

⑩ 日本国特許庁(JP)

⑪ 特許出願公開

⑫ 公開特許公報(A)

昭63-192135

⑬ Int. Cl.⁴

G 06 F 9/30
9/44

識別記号

3 5 0
3 1 0

庁内整理番号

G-7361-5B
8724-5B

⑭ 公開 昭和63年(1988)8月9日

審査請求 未請求 発明の数 6 (全30頁)

⑮ 発明の名称 信号処理方法、信号プロセッサ、その設計方法及びマイクロプロセッサ

⑯ 特 願 昭62-317311

⑰ 出 願 昭62(1987)12月15日

優先権主張 ⑱ 1986年12月15日 ⑲ 米国(U S) ⑳ 941,450

㉑ 発 明 者 グレゴリー・アラン・ アメリカ合衆国コネチカット州、エンフィールド、アーバー・ロード 50
ポータノヴァ

㉒ 発 明 者 ブライアン・ジョセ アメリカ合衆国コネチカット州、エンフィールド、マーチフ・スプラギュー ウッド・ロード 18

㉓ 出 願 人 ユナイテッド・テクノ アメリカ合衆国コネチカット州、ハートフォード、フィナロジャーズ・コーポレイ ション
ンシヤル・プラザ 1

㉔ 代 理 人 弁理士 明石 昌毅

明 細 書

1. 発明の名称

信号処理方法、信号プロセッサ、その設計方法及びマイクロプロセッサ

2. 特許請求の範囲

(1) 信号処理方法に於いて、

別々のバスを経て縮小命令集合命令及びデータを処理する過程と、

前記データバスを経て複合命令及びデータを処理することにより、またエミュレートされた各複合命令に対する縮小命令の複数の群であって各群が縮小命令の特定の複合命令に対応する縮小命令の複数の群の一つを処理することにより、複合命令集合信号処理方法をエミュレートする過程と、

を含んでいることを特徴とする信号処理方法。

(2) 複合命令集合命令に回答する信号プロセッサの設計方法に於いて、

縮小命令集合信号プロセッサとして使用するため別々のデータ及び命令バスを有する縮小命

令集合信号プロセッサを設計する過程と、

前記縮小命令集合信号プロセッサのなかへエミュレーション能力を設計する過程とを含んでおり、それにより前記データベースが到来する複合命令集合命令及びそれに関係する双方向データの双方に対して使用され、また前記命令バスが縮小命令集合命令のみを受信するために使用され、複数の縮小命令集合命令が受信された各到来複合命令に対して実行されることを特徴とする信号プロセッサの設計方法。

(3) 信号プロセッサに於いて、

縮小命令集合信号プロセッサとして使用するため別々のデータ及び命令バスを有する縮小命令集合信号プロセッサと、

前記信号プロセッサを複合命令集合命令信号と前記データベースを経て受信されたデータ信号とに回答するようにするため、また前記データベースを経てデータ信号を供給するためのエミュレート手段とを含んでおり、前記エミュレート手段が前記信号プロセッサを、受信された各複

合命令集合命令信号に回答して、前記縮小命令集合信号プロセッサ命令アドレスバス上に縮小命令アドレスを供給するようにさせ、また、供給されたこのような各縮小命令アドレスに対して、前記信号プロセッサをそのエミュレーション中に実行するための一連の対応する縮小命令を供給するようにさせることを特徴とする信号プロセッサ。

(4) 別々のデータ及び命令バスを有する信号プロセッサに於いて、

第一の入力信号及び第二の入力信号に論理演算を実行するため、また前記論理演算の結果を示すA L U出力信号を供給するため第一の入力信号及び第二の入力信号に回答する算術論理演算装置(A L U)と、

前記A L U出力信号を記憶及び供給するため前記A L U出力信号に回答するアキュムレータと、

前記A L U出力信号又はデータバスからの到来オペランド信号を選択された記憶レジスタの

に供給される例外命令アドレス信号を記憶及び供給するため前記A L U信号に回答する命令アドレスカウンタと、

前記命令信号を記憶及び供給するため命令バスからの命令信号に回答する命令レジスタと、

信号プロセッサを制御するため命令信号を復号するため前記命令レジスタからの前記命令信号に回答する制御手段と、

前記源レジスタ出力信号及び前記命令信号に、また前記第一のA L U入力信号を供給するため選択信号に回答する第一のマルチプレクサ手段と、

前記宛先レジスタ出力信号、前記アドレス信号及び前記アキュムレータからの前記A L U出力信号に、また前記第二のA L U入力信号を供給するため選択信号に回答する第二のマルチプレクサ手段と

を含んでいることを特徴とする信号プロセッサ。

(5) 別々のデータ及び命令バスと共に使用する

なかに記憶するため、宛先入力信号を供給するため、また源入力信号を供給するため、前記A L U出力信号又は前記オペランド信号に回答する、複数個の記憶レジスタを有するレジスタファイルと、

前記A L Uに論理演算の実行のために供給するため、もしくはデータバス上に出力オペランド信号として供給するため、選択された記憶レジスタの信号内容を宛先レジスタ出力信号として供給する以前に前記信号内容を記憶するため、前記宛先入力信号に回答する宛先レジスタと、

前記A L Uに論理演算の実行のために供給するため、もしくはデータバス上に出力オペランド信号として供給するため、選択された記憶レジスタの信号内容を記憶レジスタ出力信号として供給する以前に前記信号内容を記憶するため、前記源入力信号に回答する源レジスタと、

次の命令をアドレス指定するためそのアドレスをインCREMENTするためインCREMENTする信号に回答し、又は各々命令アドレスバス上

ため機械サイクルあたり少なくとも二つのクロック周期を有する本質的に一つの機械サイクルの命令集合信号プロセッサに於いて、

次の命令をアドレス指定するため内部に記憶されている現在のアドレス信号をインCREMENTするため各機械サイクルの第一のクロック周期の第一の部分の間に供給されるインCREMENTする信号に回答し、また例外命令をアドレス指定するため命令アドレスバス上の例外命令アドレス信号を供給するため内部に記憶するため各機械サイクルの第一のクロック周期の第一の部分の間にA L U出力信号に回答する命令カウンタと、

各機械サイクルの第二のクロック周期の第二の部分で開始し且つそれに続く前記命令信号を受信、記憶及び供給するため、命令カウンタにより第一のクロック周期の第一の部分の間にアドレス指定された命令信号を置換し且つ第二のクロック周期の第二の部分の開始以前に命令バス上に送るべく、第二のクロック周期の第二の

部分の間に応答する命令レジスタと、

各機械サイクルの第一のクロック周期の間に前記命令レジスタ信号を復号するため、また各機械サイクルの第二のクロック周期の間に信号プロセッサに対する制御信号を供給するため、先の機械サイクルの第二のクロック周期の第二の部分の間に前記命令レジスタのなかに記憶された前記命令信号に各機械サイクルの第一のクロック周期の間に応答する制御手段と、

論理演算を実行するため、また次の機械サイクルの第一のクロック周期の第一の部分の間の前記論理演算の結果を示すA L U出力信号を供給するため、第一の入力信号と、第二のクロック周期の第一の部分で開始した続いて供給される第二の入力信号とに選択された機械サイクルの間に応答する算術論理演算装置 (A L U) と、

前記A L U出力信号を記憶及び供給するため、選択された機械サイクルの第一のクロック周期の第一の部分の間に前記A L U出力信号に応

二のクロック周期の第一の部分で開始する宛先出力信号として前記記憶された情報を供給するため、前記宛先入力信号に応答する宛先レジスタと、

選択された記憶レジスタのなかに信号情報を記憶するため、またいずれも第二のクロック周期の第一の部分で開始して供給される、前記A L Uにより演算されるべき源出力信号もしくはオペランドアドレスバス上の出力オペランドアドレス信号として前記の記憶された源信号を供給するため、前記源入力信号に応答する源レジスタと、

第一の選択信号に応答して、第二のクロック周期の第一の部分の間に始動する前記A L Uに前記第一の入力信号として前記源出力信号もしくは前記命令信号を供給するため、選択された機械サイクルの間に、前記源レジスタにより供給される前記源出力信号と先の機械サイクルの間に記憶された前記命令信号とに応答する第一のマルチプレクサ手段と、

答するアキュムレータと、

選択された記憶レジスタのなかに前記A L U出力信号及び前記オペランド信号を記憶するため、選択された機械サイクルの第一のクロック周期の第一の部分の間に前記A L U出力信号に応答し、もしくは選択された機械サイクルの第一のクロック周期の第一の部分の間にデータバスからの到来オペランド信号に応答し、また宛先入力信号を供給するためまた記憶レジスタを選択するため第一のクロック周期の第二の部分の間に記憶レジスタを選択するため、また源入力信号を供給するため第一のクロック周期の第二の部分の間に記憶レジスタを選択するため、前記制御手段からの前記制御信号の一つ又はそれ以上に応答する、複数の記憶レジスタを有するレジスタファイルと、

選択された記憶レジスタのなかに信号情報を記憶するため、また前記A L Uにより演算されるべき宛先出力信号としてもしくはデータバス上の出力オペランド信号として使用するため第

二の選択信号に応答して、第二のクロック周期の第一の部分の間に始動する前記A L Uに前記第二の入力信号として前記アキュムレータからの前記A L U出力信号、前記宛先出力信号もしくは前記命令アドレス信号を供給するため、選択された機械サイクルの間に、前記アキュムレータからの前記A L U出力信号と前記宛先により供給される宛先出力信号と前記命令カウンタのなかに記憶された前記命令アドレス信号とに応答する第二のマルチプレクサ手段と

を含んでいることを特徴とする信号プロセッサ。

(6) 縮小命令集合マイクロプロセッサに於いて、

外部命令バスから命令信号を受信するための命令バス入力ポートと、

データ信号をそれぞれ外部データバスから受信し且つそれへ供給するためのデータポートと、

データ信号をそれぞれデータポートから探索し且つそれへ供給するため前記命令信号に応答する中央プロセッサユニットと

を含んでいることを特徴とするマイクロプロセッサ。

3. 発明の詳細な説明

技術分野

本発明は計算機、一層詳細には縮小命令集合計算機 (Reduced Instruction Set Computer) (RISC) に係る。

背景技術

複合命令集合計算機 (Complex Instruction Set Computer) (CISC) の提案者は以前にはソフトウェアのみによりなされた機能をするのに高度に複雑なマイクロプロセッサハードウェアを構成するべくますます超大規模集積回路を使用する。これはソフトウェアがますます高価になり、またハードウェア (VLSI) がますます安価になってきた結果である。より安価でより複雑なハードウェアを使用することにより、CISC設計者は、より高いレベルの言語がより簡単に、従ってまたより使用し易くなり、またソフトウェア開発費用が低下すると推論する。

正確に実行される機能により自動的に行われる。

RISC設計アプローチには多くの局面が存在し得る。種々の源から抽出されたこのような局面を説明する見事な試みがIEEE計算機、1985年9月、第8～19頁のロバート・ビー・コウエル他論文“計算機、複雑さ及び論点”になされた。そこに説明されている局面は、(1) 計算機の命令の流れを支配する簡単な機能の迅速な実行を容易にし、また低い解釈的オーバーヘッドを助長するための“単一サイクル演算”、(2) 単一サイクル演算の要望から続く“ロード/記憶設計”、(3) マイクロコードはより遅い制御経路に過ぎない解釈的オーバーヘッドを増すので、最も迅速に可能な単一サイクル演算を行うための“ハード配線による制御”、(4) 制御手段による迅速で簡単な解釈を容易にする“相対的に少数の命令及びアドレス指定モード”、(5) 一貫した使用により、再び制御経路を速くするハード配線による復号を容易にするための“固定された命令フォーマット”、及び(6) コンパイラのなか

他方に於いて、RISC提案者はより多くの機能がソフトウェアによりなされるより簡単なマイクロプロセッサを創造する。このような機能は所与のプログラムのなかで実行される機能の大多数は、全て一つの機械サイクル内での実行のために設計され得るロード、記憶、比較、分岐、整数算術、論理シフトなどのようなむしろ簡単な機能である傾向を有するという洞察に基づいている。実行されるべき各複合機能に対する特定のCISCのアーキテクチャを解釈するためのマイクロ命令を有する代わりに、RISC内のアーキテクチャ的アプローチはハードウェアによる直接実行のために設計された縮小命令集合を有していなければならない。従って、マイクロプロセッサアーキテクチャは、この場合にまさにコードであるマイクロアーキテクチャに正確にマッチしている。すなわち制御線をセットするためマイクロコード内に1及び0をセットすることを含む煩雑なプログラミングが存在しないので、必要とされる解釈は存在しない。全てのことがコード内で

へ静的実行時間を明示的に移動させる機会を提供する“より多くのコンパイル・時間努力”。

上記の文献中に指摘されているように、RISC及びCISCに関する通常の議論は、おそらくそれらの頭語のために、議論の領域が機能の命令集合の候補を選択することに制限されるべきであることである。命令の数はRISC文献中の第一の論点の一つであるけれども、RISC理論の最良の一般化はこの論点を越えている。コルウェル他は、RISC理論が、いくつかの特殊な文脈のなかで測って、性能を最大化するためアーキテクチャ/実現、ハードウェア/ソフトウェア及びコンパイル時間/実行時間の境界を横切って自由に且つ意識的に設計のトレードオフをする意志を含むことを指摘している。

この考えによれば、RISC及びCISCの頭語は任意の領域が一方又は他方として分類され得ることを意味するように見えるけれども、実際にはRISC及びCISCは連続的な多次元設計空間の異なるコーナーにあるものとして考えられ

るべきである。従って、他方と相容れない一方が必要とされるのではなく、新しいシステムを想像するのに設計者により使用され得るようにCISCの経験及びRISCの信奉から引き出された技術の組合わせを定型化することが必要とされる。

上記のことにもかかわらず、命令の数がアーキテクチャをRISC又はCISCとして分類するための重要な規範であることは理解されよう。従って、かかる技術を定型化する以下のRISCの開示及び特許請求の範囲は、コルウェル他により説明されているようなRISC機械の属性のいくつか又は全てを有するRISCに制限することなしに、実際に縮小命令集合を有する計算機に対する設計フィロソフィに属するものとして理解されるべきである。換言すれば、以下の開示はいくつかのRISC構想を引いてはいるが、“設計空間”の“コーナー”にはない。

軍は航空機計算機に対して標準16ビット複合命令集合アーキテクチャ(MIL-STD-1750及びその後継)を制定してきた。この標準

ソフトウェアの使用及び再使用である。他の利点として、(a) 武器システム内の二つ又はそれ以上の計算機の使用により得られる全支援ソフトウェアの減少、及び(b) ハードウェアの開発と無関係なソフトウェアの開発のような利点も達成される。

MILはプログラムの観点からの機械的操作を規定する。それはデータ形式、命令形式、簡略命令、命令演算、アドレス指定モード、プログラマアクセスレジスタ、割込み構造などを規定する。それは特別な実現を規定しないので、それはベンダー及びテクノロジーに無関係である。上記のように、その使用は標準ソフトウェア支援ツールの使用、マルチベンダー軍用システム中の全支援ソフトウェアの減少、及びハードウェアの開発と無関係なソフトウェアの開発を助長するものと期待される。

MIL-STD-1750レジスタ集合は16の16ビット汎用レジスタ(R0、...、RF)、16ビット状態語(SW)、16ビット命令カウン

の目的は、計算機の特殊な実現の詳細を定めることなしに空軍の航空兵器システムの仕様を定めるための均一な命令集合アーキテクチャを確立することである。従って、それは複合命令集合アーキテクチャを規定するのみであり、システム独特の要求は各特殊の計算機に対する後日の規定のために残されている。その応用は特定の航空用機械又は特殊のハードウェアに制限されていない。一般に、この標準は適度な精度の航行、計算されたエアーレーズ点、武器供給、エアーランデブー、貯蔵管理、航空機ガイダンス及び航空機管理のような機能を実行する計算機に、制限なしに、応用可能である。記述“MIL-STD-1750”はこの文書を通じて、特殊な改訂版が記述されており従ってその意味で理解されるべき場合を除いて、最初の標準及びその後継の全てを記述するのに使用され得る。

MIL-STD-1750標準の命令集合アーキテクチャの期待される利点はコンパイラ及び命令レベルシミュレータのような利用可能な支援

タ(IC)、16ビットマスクレジスタ(MK)、16ビット割込みレジスタ(P1)及び16ビット故障レジスタ(FT)を含んでいる。

支援されるデータフォーマットはバイト(上側、下側)、16ビット固定点単精度(16ビットの2の補数)、32ビット固定点倍精度(32ビットの2の補数)、32ビット浮動点(24ビットの2の補数仮数、8ビットの2の補数指数)及び48浮動点拡張精度(48ビットの2の補数、8ビットの2の補数指数)を含んでいる。

MIL-STD-1750命令集合はこれまでにCISCを使用していくつかの会社により実現されてきた複合命令集合である。例えば、なканずくフェアチャイルド、マクドネル・ダグラス及びパーフォーマンス・セミコンダクターは全て市販されているMIL-STD-1750CISC機械を有する。

発明の開示

本発明の目的は、簡単なマイクロプロセッサを提供することである。提供されるマイクロプロセ

ッサは縮小命令集合計算機 (RISC) として、又は簡単に簡単なアーキテクチャを有する信号プロセッサとして見られ得る。"RISC" という用語は本発明のこの目的と結び付いて本明細書を通じて頻りに使用される。

本発明の他の目的は、たいていの命令に対して単一サイクル演算を有する縮小命令集合計算機 (RISC) を提供することである。

本発明の他の目的は、単一サイクル演算の上記の目的に付随する RISC ロード/記憶設計を提供することである。

本発明の他の目的は、迅速な単一サイクル演算のためのハード配線による制御を提供することである。

本発明の他の目的は、制御手段による迅速で簡単な解釈を容易にする比較的少数の命令及びアドレス指定モードを提供することである。

本発明の他の目的は、制御経路を速くするため命令のハード配線による復号を容易にするのに一貫して使用され得る固定されたフォーマットを提

合信号プロセッサを提供することである。

本発明の他の目的は、複合命令に回答するが、それらを縮小命令の群を使用して実行する信号プロセッサの設計方法を提供することである。

本発明の他の目的は、縮小命令集合信号プロセッサのアーキテクチャがハードウェア ("シリコン") に設計され且つ縮小され得ると共に (縮小命令のなかの) 複合命令集合のコードエミュレーションのためのプログラミングが同時に行われる信号プロセッサを迅速に設計するための方法を提供することである。換言すれば、本発明のこの目的は、信号プロセッサが迅速な設計サイクルを有する複合命令集合をエミュレートするための縮小命令集合信号プロセッサを提供することである。

本発明の第一の局面によれば、ハーバード・アーキテクチャを有する、すなわち別々のデータ及び命令バスを有する縮小命令集合計算機 (RISC) が提供される。

本発明の第二の局面によれば、RISC が複合

供することである。

本発明の他の目的は、多数の可能な縮小命令集合で使用可能な簡単な RISC アーキテクチャを提供することである。

本発明の他の目的は、このような簡単な RISC アーキテクチャの実現に迫る縮小命令集合を提供することである。

本発明の他の目的は、複合命令集合をエミュレートするため縮小命令集合を使用する信号処理方法を提供することである。

本発明の他の目的は、MIL-STD-1750 をエミュレートし得る縮小命令集合信号プロセッサを提供することである。

本発明の他の目的は、ユーザーがオフチップの記憶及び呼出しのための RISC コードサブルーチンを書くことにより機能を定義し得る MIL-STD-1750 マイクロプロセッサを提供することである。

本発明の他の目的は、複合命令集合を効率的にエミュレートするのに使用するための縮小命令集

合命令集合計算機 (CISC) のエミュレータとして使用するため設計されている。RISC はそのオペランド又はデータバスを経て受信された複合命令に、RISC 命令の各々複合命令の一つに対応する複数個の群の対応する一つを、RISC 命令バスを経て、アドレス指定することにより、回答する。いったん受信された複合命令に対応する群の第一の命令がアドレス指定され且つ実行されると、特定の群のなかの組み合わせられている縮小命令の残余は、群のなかの命令の全てが完全に実行され終わるまで、順次にアドレス指定され且つ実行され得る。こうして、複合命令が縮小命令の群を使用してハードウェアのなかでエミュレートされる。縮小命令は RISC 機械の命令バスを経て受信される。

さらに本発明のこの第二の局面によれば、ハーバード・アーキテクチャを有する縮小命令集合計算機が、設計される。複合命令集合計算機をエミュレートするための縮小命令集合信号プロセッサとしてだけでなく、以前のように RISC 命令

に対する命令バスを使用して但しこのモードではオペランドに対してのみデータバスを使用して縮小命令集合動作モードで動作する縮小命令集合信号プロセッサとしても使用するために設計される。これはRISC動作モードとして特徴付けられ得る。希望によりRISC又は複合命令エミュレーションモードが選択され得る。

さらに本発明の第二の局面によれば、縮小命令の複数個の群が信号プロセッサの外部のメモリストアのなかに記憶される。上記のように、群のなかの命令は実行のために順次に記憶される。換言すれば、群のなかの第一の縮小命令のアドレス指定は群のなかの他の縮小命令の各々の順次の実行により迅速に追隨される。

さらに本発明の第二の局面によれば、エミュレートされる複合命令集合はMIL-STD-1750命令集合である。

本発明の第三の局面によれば、複合命令集合命令に回答する信号プロセッサの設計方法は、第一に、縮小命令集合信号プロセッサとして使用する

ため別々のデータ及び命令バスを有する縮小命令集合信号プロセッサを設計する過程と、第二に、複合命令集合をエミュレートするべく縮小命令集合信号プロセッサを設計する過程とを含んでおり、それによりデータバスが到来する複合命令集合命令及びそれに関係する双方向データの双方に対して使用される。RISC命令バスは縮小命令集合命令のみをアドレス指定且つ受信するために使用される。複数個の縮小命令集合命令が受信された各到来複合命令に対して実行される。こうして、ハードウェアが、エミュレーションコードが同時に書かれるている間に実行するべく設計且つ縮小され得る。

さらに本発明の第一の局面によれば、信号プロセッサアーキテクチャは、第一の入力信号及び第二の入力信号に論理演算を実行するため、また前記論理演算の結果を示すALU出力信号を供給するため第一の入力信号及び第二の入力信号に回答する算術論理演算装置(ALU)を有する。信号プロセッサアーキテクチャは、ALU出力信

号を記憶し且つ"B"マルチプレクサ(BMUX)に供給するためALU出力信号に回答するアキュムレータをも有する。"B"マルチプレクサは第二のALU入力信号を供給するため、アキュムレータ出力信号を含む三つの異なる信号のなかから一つを選択する。BMUXへの入力の他の一つは、ALUにより実行又は走査されるべきアドレスを成すRISC命令カウンタ出力信号である。BMUXへの第三の入力は、レジスタファイルのなかの多数のレジスタの任意の一つに回答する宛先レジスタの出力である。宛先レジスタ出力信号が代わりにオペランドバス上にデータとして供給され得る。命令カウンタはALUから始動命令アドレスを受信し、またアドレスを順次にインクリメントするため規則的にクロックされ得る。ALU出力信号はデータ又はオペランドバスからの到来オペランド信号に回答する多数のレジスタを含むレジスタファイルにも供給される。レジスタファイルは到来命令から復号されたものとして選択された記憶レジスタのなかにALU出力信号もし

くはオペランド信号を記憶する。レジスタファイルは宛先レジスタ及び源レジスタにレジスタを供給する。源レジスタはレジスタファイルから受信された信号を記憶し、またそれを"A"マルチプレクサ(AMUX)に又は出力としてオペランドアドレスバスに供給する。AMUXはRISC命令カウンタにより先にアドレス指定されたRISC命令バスから受信されたRISC命令にも回答する。AMUXはALUに第一の入力信号を供給する。ハード配線による制御装置は到来する複合命令を復号し、また適当な順序で上記のアーキテクチャを作動させるために必要な制御信号の全てを供給する。

ハード配線による制御装置は、(i)各機械サイクルの第一の四分の一の間にRISC命令カウンタにインクリメントする信号を供給することにより順次に記憶された命令をアドレス指定するため、(ii)選択された機械サイクルの第二の四分の一の間に、ALUにより演算されるそれらの信号内容を有するようにレジスタファイルのなかの

記憶レジスタを選択するため、また源及び宛先レジスタのなかに記憶するため選択されたレジスタ信号内容を第二の四分の一の間に供給するため、レジスタファイルに制御信号を供給するため、(iii) 選択された機械サイクルの第二の四分の一の間に、メモリからレジスタファイルへ又はレジスタファイルからメモリへデータをロード又は記憶するべく命令に回答してオペランドアドレス及びデータバスをイネーブルするためのイネーブル信号を供給するため、(iv) 選択された機械サイクルの第三の四分の一の間に開始して、ALUに対する前記第一の入力信号として供給するため選出力信号と命令信号との間を選択するためAMUXに第一の選択信号を供給するため、(v) 選択された機械サイクルの第三の四分の一の間に開始して、ALUに対する第二の入力信号として供給するため宛先出力信号とアキュムレータ出力信号とと命令アドレス信号との間を選択するためBMUXに第二の選択信号を供給するため、(vi) 選択された機械サイクルの第二の四分の一の間に開始

は、それぞれ複合命令集合のなかの複合命令の一つをエミュレートするために設計されている複数個のかかる群の一つである。複合命令集合プログラムアドレスカウンタは、順次にアプリケーションのなかの次の複合命令をアドレス指定するため、インCREMENTする信号に回答し、又は例外複合命令アドレス信号を記憶及び供給するためALU出力信号に回答する。

さらに本発明の第二の局面によれば、受信された複合命令信号は制御手段により復号され、また選択された機械サイクルの第一の四分の一の間にインCREMENTする信号をプログラムカウンタに供給することにより順次にアドレス指定される。制御手段は選択された機械サイクルの第三の四分の一の間にRISC命令アドレスカウンタへのかかる縮小命令の群を開始するため縮小命令集合アドレス信号をも供給する。また制御手段は複合命令信号を受信、記憶及び供給するべく選択された機械サイクルの第一の四分の一の間に複合プログラムレジスタをイネーブルする。

して、ALUにALU演算選択信号を供給することによりALUにより実行されるべき演算を選択するため、(vii) 選択された機械サイクルの第一の四分の一の間に開始して、適当なレジスタにALU出力選択信号を供給することによりレジスタファイル、アキュムレータ又は命令カウンタのなかにALU出力信号を記憶するため、(viii) 選択された機械サイクルの延長された第四の四分の一の間に、シフト、乗算及び除算を実行するためシフト信号を供給するため命令信号を復号する。

さらに本発明の第二の局面によれば、かかるRISC信号プログラムはさらに、順次に記憶された縮小命令集合信号の群の第一の命令信号をアドレス指定するため縮小命令集合アドレス信号を復号し且つRISC命令カウンタに供給するためかかる複合命令信号を記憶し且つ制御手段に供給するためデータバスを経て受信される複合命令で書かれた応用プログラムに關係して供給される複合命令に回答する複合命令集合プログラムレジスタを含んでいる。アドレス指定されるかかる群の各

本発明の第一の局面によれば、CISCのエミュレーションをなんら参照することなしに、RISC機械として簡単に設計及び使用され得る簡単なアーキテクチャが提供される。もちろん、背景技術のところで説明したように、かかる簡単なアーキテクチャは同時にRISC属性及びCISC属性を有し得るので、二つの間のハードな速い境界を作ることは困難である。従って、本アーキテクチャは二つの極限の間の設計空間に置かれ、上記の広い「縮小命令」の意味で例外として実際に厳密に特徴付け可能でないことは理解されよう。本明細書中に開示される「縮小」命令集合は、ここに開示されるアーキテクチャで実行するために特に良く適している。しかし、他のRISC命令集合がここに開示されるアーキテクチャでの実行のために定型化され得ることは理解されよう。

それにもかかわらず、提供される簡単なアーキテクチャは、制限なしに、以下に説明されるいくつかの異なるRISCアーキテクチャ・アプ

ローチをとる。第一に、シフト、乗算及び除算命令を除く全ての命令は単一の“2クロック”機械サイクルのなかで実行される。オペランドはロード及び記憶アクセスメモリのみを有するレジスタ・ツー・レジスタである。これは内部制御を簡単化する。全てのRISC命令はハード配線による制御を使用して実行される。マイクロコードは使用されない。32命令演算のみが実行される。アドレス指定はレジスタ間接及び即値モードに制限されている。命令フォーマットは簡単であり、また境界を横切らない。

ここに開示されるRISCアーキテクチャに加えて、信号プロセッサが複合命令集合から受信された複合命令に実際に応答する意味でCISCがエミュレートされ得る、本発明の第三の局面による、特に有用な設計方法が開示される。RISC機械により受信されたこのような各複合命令は、エミュレーションによりこのような各命令を実行するため予めエミュレートされた縮小命令の特定の群のアドレス指定をトリガする。こうして、

ハードウェアが同時に縮小されるので、迅速なターンアラウンドの利点を提供する。しかし、速度は唯一の利点ではない。いったんハードウェア設計が“シリコン”に縮小されると、設計プロセスはそれに関して完了され、その後の変更はなされ得ない。設計エラーはたいていこの段階では必然的に発見され、他の費用のかかる設計サイクルを必要とし、第二の設計の後にも追加的なエラーが発見される可能性がある。これは時間及び金の点で非常に費用のかかるプロセスである。本発明は、ハードウェア設計“問題”を受け入れるけれども、それにもかかわらずハードウェア問題を迂回するべくエミュレーションコードを使用する補正を受け得るハードウェアのなかに非常に簡単なアーキテクチャを設計することにより、この問題の回避を許す。当業者に知られているように、設計プロセスの間に、機能は通常は一つよりも多い仕方でも、所与の信号プロセッサのなかで、設計者により影響され得る。こうして、もしRISC機械を使用してCISCをエミュレートすることを試みる

エミュレートされた各複合命令に対して、ハードウェア内での呼出し及び実行のために記憶された縮小命令の群が存在する。この設計アプローチはRISCアーキテクチャがハードウェア実施例に対して迅速に、すなわち同時にエミュレーションコードが開発されている間にゲートアレーとして設計されることを許す。こうして、もし設計者がCISCの設計を望むならば、設計者は、同時に複合命令集合をエミュレートするためエミュレーションコードの開発を続行しつつ、先ずRISCを設計し、続いてそれを“シリコン”に縮小することにより目的物を一層迅速に得ることが出来る。このアプローチの費用は純粋なCISCとしてCISCを設計する費用の10%よりも少なく、且つ優れた結果が得られると見做られる。主な費用節減の一つは、いくつかの設計サイクルの形態でCISC設計者により通常経験される設計リスクに関係付けられる。

この設計アプローチは、同時にエミュレーションコードが書かれている間に“シリコン”にハ

ードウェアが同時に縮小されるので、迅速なターンアラウンドの利点を提供する。しかし、速度は唯一の利点ではない。いったんハードウェア設計が“シリコン”に縮小されると、設計プロセスはそれに関して完了され、その後の変更はなされ得ない。設計エラーはたいていこの段階では必然的に発見され、他の費用のかかる設計サイクルを必要とし、第二の設計の後にも追加的なエラーが発見される可能性がある。これは時間及び金の点で非常に費用のかかるプロセスである。本発明は、ハードウェア設計“問題”を受け入れるけれども、それにもかかわらずハードウェア問題を迂回するべくエミュレーションコードを使用する補正を受け得るハードウェアのなかに非常に簡単なアーキテクチャを設計することにより、この問題の回避を許す。当業者に知られているように、設計プロセスの間に、機能は通常は一つよりも多い仕方でも、所与の信号プロセッサのなかで、設計者により影響され得る。こうして、もしRISC機械を使用してCISCをエミュレートすることを試みる

ならば、RISCを設計し、また、最初の設計のなかで最初に意図された仕方でも複合命令が実行されるのを許さないハードウェア設計のなかの“欠陥”が存在する場合には、設計者は他の仕方でも“問題”複合命令をエミュレートするべくエミュレーションコードのまわりを変更し得る。こうして、本アプローチはハードウェア設計欠陥を許容するべくフレキシビリティを提供する。

ここに開示される、RISCを使用する複合命令集合計算機を実現するための設計方法はハードウェアとソフトウェアとの間にCISCを設計するタスクを分割する。設計効率はソフトウェアの効率から来る。従来の技術では、全ての機能は一つのチップ上にあった。本開示は、制御機能が実行機能から分離されている“2チップ解決”の使用を教示する。ここに開示される簡単な機械はハードウェア設計プロセスを速くし、またエミュレーションコードの同時設計を許す。従来の技術で経験された設計リスク、すなわち正しく作動する設計を得る以前にいくつかのハードウェア繰り返

しにさらされるリスクは回避される。

本発明の第二の局面は、CISCをエミュレートするため群のなかに配置されたRISCコードを実行するRISC機構を提供する。この局面の実施例は、純粋なRISCモードでハーバード・アーキテクチャを有する純粋なRISC機構として、もしくはCISCをエミュレートするため群のなかに配置されたRISCコードを実行するRISC機構として、RISC機構をランさせる可能性を提供し得る。いずれの場合にも、コードのこのような各群は、対応する複合命令に回答して実行され得る。いったん群の初期アドレスがRISCアドレスバス上に置かれると、その群のなかの縮小命令の他のメンバーが順次に実行される。RISCコードの群によりエミュレートされるCISCを有する新規なアプローチと、まさにRISCとして又はエミュレータとしてランするRISC機構を有する新規なアプローチとは縮小命令集合のフィロソフィによる強力な信号処理ツールを提供し、また縮小命令集合アーキテクチャ

める可能性を与える。これらは、RISC機構と組み合わせられており、またRISCマイクロプロセッサと共に使用するためのPROMのようなメモリ装置のなかに別々に記憶されているRISC命令集合を使用して、簡単に顧客により設計され得る。完全なRISC命令集合はもちろん、ユーザーにより定められた機能を顧客が書くことを可能にするべくMIL-STD-1750マイクロプロセッサの購入時に供給される製品文書のなかで顧客に説明されなければならない。

本発明の第一の局面の以上の教示の全てに追加して、特殊な縮小命令集合が、前記のように、その命令集合及び多くの他の可能な類似の命令集合を実現するために特に有用である特殊なRISCアーキテクチャとならんで教示される。開示される特殊なRISCアーキテクチャは後で開示されるような第二の局面によりMIL-STD-1750命令をエミュレートする目的にも非常に有用である。

本発明の前記及び他の目的、特徴及び利点は以

て使用する複合命令集合をエミュレートする可能性を同時に提供する。本発明のこの局面は、こうして使用される時、まさに強力なツールである。

RISCコードのなかで実行されるべきユーザー固有の、ユーザーにより定められた機能をユーザーが開発し得ることは、本発明の第二の局面により教示される縮小命令集合信号プロセッサの他の強力な特徴である。通常、MIL-STD-1750によるCISCの製造者は、もしその顧客がそのMILにより許されるようなユーザーにより定められた機能を実現することを望むならば、その顧客が特注の特別に開発されたMIL-STD-1750によるマイクロプロセッサチップを注文することを要求する。(MIL-STD-1750Aはユーザーにより定められた機能を暗示的に許し、他方に於いてMIL-STD-1750Bはこのような機能を明示的に準備する)。本発明はユーザーに、チップが購入された後にRISCコードのなかでの実行のために任意の数のこのようなユーザーにより定められた複合命令を定

下にその好ましい実施例を図面により詳細に説明するなかで一層明らかになろう。

発明を実施するための最良の形態

第1図には、縮小命令集合計算機(RISC)として広く特徴付けられてよく、また通常は、制限なしに、マイクロプロセッサの形態をとる、本発明の第一の局面による簡単な信号プロセッサ10がブロック図で示されている。本発明の新規なRISCはハーバード・アーキテクチャを有する。すなわち命令及びデータバスが別々である。RISC命令アドレスバス12は縮小命令をアドレス指定するのに使用されており、縮小命令は次いでRISC10により縮小命令バス14を経て受信される。オペランドアドレスバス16は双方向であってよいデータバス18を経てRISC10により受信もしくは供給されるデータをアドレス指定するのに使用される。背景技術のところに記載したように、頭語「RISC」はここでは縮小命令集合機構の広い意味で使用されている。

ハーバード・アーキテクチャの使用は、ノイ

マン・アーキテクチャが好ましいアプローチであった以前のCISCに使用されているような多重化命令/データバスとは異なっている。ハーバード・アーキテクチャによるアプローチは、命令及びデータが同時にアクセスされ得るという事実により、RISCがはるかに速い速度で動作することを許す。これは性能を最大化する上記の一般的RISCアプローチと調和している。

第2図は第1図のRISCと類似しておりまた同一のハーバード・アーキテクチャを有するRISC10のブロック図である。第2図のRISC10は、共に使用される時に、本発明の第二の局面に従って、複合命令集合計算機(CISC)をエミュレートし得るプログラマブル・リードオンリメモリ(PROM)のようなメモリ装置20と組み合わされている。RISC10 PROM20は共にCISCをエミュレートするべくRISCを使用する構想の“2チップ”22の実施例又はそれへの“解決”として考えられ得る。2チップ22は実際にCISCである。これは設計者

るためRISC命令アドレスバス12上のアドレスをインCREMENTする。このプロセスは、(データバス10を経て受信された特定のCISC命令をエミュレートするためPROM20のなかに記憶されている)RISC命令の全ての群がエミュレーションを経由して実行され終わるまで継続する。所与の命令は、なにかんずく、データバス18及びその(単にデータ線又は宛先への経路を示す)延長部26を経由してオペランドをロード又は記憶するのに使用されるメモリロード/記憶操作を含んでいる。

RISC

第3図には、第1図及び第2図中に示されているRISCに類似のRISC10の内部の簡略化されたブロック図が示されている。第3図の細線は、本発明のCISCエミュレーションの局面なしに、第1図のRISC10に一致する。先ず、本発明の第一の局面のRISC実施例の動作を完全に説明するため、第3図を先ずこれらの細線の部分に関して説明する。次いで、本発明の第二の

及びユーザーの双方の観点から上記のように高度に効率的なアプローチである。

信号の形態の複合命令は信号線24を経てデータバスに供給され、それによりRISC10に伝達される。複合命令24の源は本発明にとって重要ではないが、一般に“CISC”22により実行されるべきいくつかの応用プログラムから成っている。複合命令はこの仕方次第に受信され、またRISC10に復号される。受信されるこのような各複合命令に対して、初期始動アドレスがRISC命令アドレスバス12を経てPROM20に供給され、そこにエミュレートされている特定のCISC命令に対応するRISC命令の群が順次に記憶される。PROM20のなかでアドレス指定された第一のRISC命令は次いで、典型的に全機械サイクルの間、RISC命令バス14上に置かれる。命令が記憶又は実行のためにRISC10に安全に供給され終わった後、RISC10は次に、命令バス14上に置くため次に順次に記憶されたエミュレーション命令をアクセスす

局面の実施例、すなわち第2図に示されているようにCISCをエミュレートするのに使用されるRISCを完全に説明するため、太線の部分を説明する。

制御装置30はRISC10に対する(図面を簡単にするため図示されていない)内部制御信号を供給する。このような制御信号により制御される機能のなかで次のRISC命令の取出しが実行されなければならない。これは、RISC命令アドレスを保持するのに使用される命令カウンタ32を、そのアドレスに記憶されているRISC命令を命令レジスタ34のなかへロードするため、インCREMENTすることにより成就される。命令レジスタ34は実際には二つのレジスタ、すなわち(現在の機械サイクルの間に)命令バス14上でRISC命令を最初に受信及び記憶するための“パイプ”又は一次レジスタと、現在の機械サイクルの中央まで先の機械サイクルの間に命令バス14上にあった命令を保持するための二次レジスタとから成っている。パイプはその内容を、後で

一層詳細に説明するように、各機械サイクルの中央で二次レジスタに伝達する。

次に第4図を参照すると、制御装置30により与えられる種々の制御信号に相当する波形が示されている。図示の目的で、第4図の種々の波形は四分の一機械サイクル40、42、44、46に分割されている単一機械サイクル38のなかで相互に比較するため時間的に“凍結”されている。任意の選択された周波数又はデューティサイクルであってよい基本システム“クロック”波形は第4図(a)に示されている。第4図(b)には後で一層詳細に説明する目的で有用である第一の四分の一40のクロック信号波形50(CK1)が示されている。第4図(c)には各機械サイクル第二の四分の一42の間に再び生ずる(この場合、高い)波形52(CK2)が示されている。第4図(d)には各機械サイクル第三の四分の一44の間に再び生ずる波形54(CK3)が示されている。第4図(e)には各機械サイクル第四の四分の一46の間に再び生ずる波形56(CK4

)が示されている。

第4図(f)には、命令レジスタ34のなかの“パイプ”レジスタのローディングのタイミングを示す波形58が示されている。波形58は、第4図(b)の波形50のCK1の立ち上がり縁と同時に生起する低レベルから高レベルへの電圧変化の立ち上がり縁に小さい矢印を付して示されている。これは、第3図のRISC命令線14上に存在する(全機械サイクルの間、その時点に存在するであろう)命令がいま安定しており命令レジスタ34のなかのRISC命令レジスタパイプのなかへのクロッキングの準備ができていていることを示す。こうして、第3図中に示されているクロック線60は各機械サイクルの開始時にCK1の立ち上がり縁に上がる。換言すれば、命令レジスタパイプの内容が各機械サイクルの開始時に変更される。命令が二つの全四分の一サイクルにわたり“パイプ”レジスタのなかに存在した後、命令は第4図(g)中の波形64の立ち上がり縁62により示されているようなCK3の開始時に“パイ

プ”から二次命令レジスタのなかへラッチされる。もちろん、第二の四分の一42の間に二次命令レジスタの入力端に与えられる信号が、クロックが低レベルであるので、この第二の四分の一42の間に二次命令レジスタの出力端にも存在していることは理解されよう。従って、まだラッチされていないとしても、この第二の四分の一周期の間に制御装置による命令コードの復号が開始され得る。こうして、“パイプ”のなかの命令は機械サイクルの開始時から終了時までそのなかに残っているけれども、パイプの内容は各機械サイクルの中央で二次命令レジスタのなかへコピーされる。従って、所与の命令が二次レジスタのなかに存在する時間は一つの機械サイクルの中央から他の機械サイクルの中央まで亘っている。

命令バス14上の命令が命令レジスタ34“パイプ”のなかへクロックされる時点で、命令アドレスバス12上のアドレスが、各機械サイクルの開始時にCK1の立ち上がり縁と一致するその作動クロッキング(立ち上がり縁70)タイミング

を有する第4図(h)の波形68に示されているタイミングの制御信号により開始されて次の順次アドレスへ通常インCREMENTされる。しかし、もし例外命令が実行されるべきであれば、それは順次アドレスの外でロードされる。

制御装置30の他の機能は、種々の仕方で、命令レジスタ34“パイプ”内及び命令レジスタ34の二次レジスタ内に留まっている命令を復号することである。各機械サイクルの第一の四分の一の間に、制御装置30は命令レジスタ34のパイプレジスタのなかへまさにロードされた命令のレジスタフィールド部分を復号する。線72はレジスタフィールドを復号するためのパイプから制御装置30への情報の転送を示す。(RISC命令フォーマットは後で一層詳細に説明する)。ここでは、示されている実施例に対して、16ビットRISCレジスタ・ツー・レジスタ命令フォーマットに対して、6個の最上位ビットが命令コードであり、次の5個の最上位ビットが宛先フィールドであり、また最後の5個のビットが源フィールド

ドであると言えば十分である。復号されたレジスタフィールド情報は、パイプのなかで、宛先レジスタ76及び源レジスタ78のなかへロードされるべきレジスタファイル74のなかのレジスタの選択された対を同定するのに使用され得る。いったんレジスタフィールドが復号されると、機械サイクルの第二の四分の一42の間に能動化される選択線80により示されているように、レジスタの適当な対が選択される。第4図(1)及び第4図(1)には、それぞれの立ち上がり縁86、88により、レジスタファイル74のなかの選択されたレジスタがその時点で、すなわち機械サイクルの中央で宛先及び源レジスタ76、78のなかへクロックされることを示す波形82、84が示されている。

この時点で、上記のように、パイプのなかの命令が命令レジスタ34のなかの二次レジスタのなかへラッチされる。命令レジスタの二次レジスタのなかで命令コードが、実行されるべき命令を決定する目的で復号される。宛先フィールドもこの

TD・1750実施例と結び付いて本開示の技術のレベルに関しては重要であり、従ってその目的で第3図に含められている)。

もし制御装置30が、命令レジスタ34のなかの二次レジスタから復号されたものとして、レジスタファイル74のなかのレジスタ又はレジスタ対のなかへ信号線100上のALU出力信号を含むALU104に対する演算を選択すれば、ALUの出力は第4図(k)の波形114の立ち上がり縁112により示されているように機械サイクルの第二の四分の一の開始時にレジスタファイル74のなかの選択されたレジスタ又はレジスタ対のなかへクロックされる。この波形はALU104により書込まれるべきレジスタをクロックするため第3図中に示されているクロック線116上に存在する電圧を表す。

こうして、各機械サイクルの第一の四分の一42の間に、制御装置30は宛先及び源レジスタ76、78をロードするためアクセスされるべきレジスタファイルのなかのレジスタの対を決定する

時点で再び、内部演算の結果もしくはデータバス18から移動されたオペランドにより書かれるべきレジスタファイル74のなかのレジスタを決定する目的で復号される。これらの二つの代替的なローディング経路は信号線100及び信号線102により示されている。信号線100はなかななく宛先及び源レジスタ76、78により与えられる信号を演算する算術論理演算装置(ALU)104の32ビット出力を表す。宛先及び源レジスタ76、78の内容がALU104により演算される演算に対して、その演算の出力は線100を経て、宛先レジスタを最初にロードしたレジスタ対のなかへ戻される。他方、情報はバッファ106、信号線108、入力/出力(I/O)装置110及び信号線102を経てデータバス18から与えられるものであってもよい。I/O装置は、これまでに説明した目的で、簡単に信号線108から信号線102への短い回路と考えられてよい。すなわち、それは本開示の現在のレベルに重要な機能はしない。(しかし、本発明のMIL-S

べく命令レジスタ34パイプを復号し、またレジスタファイル74のなかのどのレジスタが入力バスで又は以前の機械サイクルからのALUの出力でロードされるべきかを決定する目的で命令レジスタ34のなかの二次レジスタを復号する。線100上のALU出力信号又は線102上のデータはCK2の立ち上がり縁で選択されたレジスタ又はレジスタ対のなかへロードされ、他方宛先及び源レジスタはCK3の立ち上がり縁でロードされる。

もし命令レジスタ34のなかの二次レジスタのなかの命令から復号された命令コードが外部メモリ空間からデータをロードするため又はそれにデータを記憶するためのロード/記憶命令であることが判明すると、信号線118及びバッファ120を経て源レジスタ78からオペランドアドレスバス16上へオペランドアドレス出力が与えられなければならない。制御装置30はもちろん、線118上の信号をオペランドアドレスバス16に通す目的で、バッファ120をイネーブルする。

これは信号線117a及び117b上でクロックされるCK3の立ち上がり縁で生起する。ロード操作時に、オペランドは次いでメモリからデータバス18上に現れ、制御装置30によりイネーブルされたバッファ106と線108と1/0110と信号線102とを経てレジスタファイル74のなかへ送られる。命令レジスタ34のなかの二次レジスタのなかに留まっているRISC命令の宛先フィールドは、それがALU演算に関する命令に対するものであったかのように、オペランドがロードされるべきレジスタファイル74のなかのレジスタを指定するために使用される。再び、これは第4図(k)の波形114に示されているようにCK2の立ち上がり縁で生起する。記憶操作時に、オペランドは、宛先レジスタ76のなかへロードされる命令の宛先フィールドにより指示されたレジスタからデータバス18上に供給される。

ここまでは、制御装置30は、命令カウンタ32をインクリメントすることによりRISC命令

マルチプレクサへのいくつかの異なる入力信号のなかから選択線132、134により選択される。選択線は二次命令レジスタを復号することにより用意され、またCK3の立ち上がり縁の生起時に存在している。換言すれば、宛先及び二次レジスタ76、78が選択されたレジスタでロードされる時、マルチプレクサが選択される。もし宛先及び二次レジスタ76、78がそれらの内容をALU104により演算するために選択されるならば、それらの内容が適当な演算を実行するため線118及び135を経てまたそれぞれのマルチプレクサ124、126を通じてALUの適当な入力端に転送される。制御装置30は命令コードが二次命令レジスタのなかに留まる間に命令コードを復号することによりALUにより実行されるべき演算を選択する。これは通常CK3の立ち上がり縁の少し後に生起する。ALUは加算、アンド、オア及び排他的オアを含む標準レパートリの演算を行い得る。いったんマルチプレクサの出力が安定化すると、演算は安定な出力を線100上に与え

を取り出し、レジスタファイル74のなかで演算されるべきレジスタを選択し且つそれらを源及び宛先レジスタ76、78のなかへ記憶することを含めて、RISC機械のなかで種々の機能を実行するため命令レジスタ34の一次("パイプ")レジスタ及び二次レジスタの双方のなかに受信された現在の命令を復号し、またレジスタファイル74のなかの選択されたレジスタのなかへ先の命令からのALU出力信号をロードし、もしくは復号されたオペランドがメモリからのロード操作を示すならばデータバス18からのオペランドをロードするその機能を実行するものとして説明されてきた。

説明されるべき次の機能は、マルチプレクサ124、126の対を通じてのALU104への入力の選択である。第一のマルチプレクサ124は第一の入力信号128をALU104の第一の入力端に与える。第二のマルチプレクサ126は線130上の第二の入力信号128をALU104の第二の入力端に与える。これらの入力信号は各

るべくハードウェア内で安定化される。

第3図中に示されているRISC10に対して、本開示の現在のレベルで、すなわち本発明の第一の局面と結び付いて、マルチプレクサ124、126の各々が唯二つの別々の信号にตอบสนองする。追加的な信号は本発明の第二の局面を説明するため本開示の他のレベルと結び付いて後で説明される。マルチプレクサの他の入力は、開示のRISCレベルに対して、RISC命令及び命令アドレスバスと結び付けられている。こうして、マルチプレクサ124は制御装置30からの線126上の命令信号にตอบสนองする。これは、RISC命令上で演算を実行することが望まれた特定の機械サイクルの第三の四分の一44の間にマルチプレクサ124のなかへ転送される命令レジスタ34の二次レジスタのなかに留まっている即値データフィールドに一致する。しかし、これらは例えば、(命令のなかの)即値データフィールドよりもむしろデータが操作される通常の内部演算に比較して大きな間隔をおいている。レジスタ・ツリー・即値

命令は後でRISC命令集合と結び付けて説明される。マルチプレクサ126は、開示のこのレベルで、命令アドレスバス12からのBISC命令アドレス信号にも応答する。このようなアドレスは時々操作され得る。こうして、制御装置30は、命令コードに従って、ALUのなかでの演算のためにマルチプレクサ124、126を通じてALUへの入力を選択する。制御装置30は、命令コードに従って、ALUにより入力信号上で実行されるべき特定の演算をも選択する。演算は、入力が安定になった後に線100上に安定な出力信号を生ずるハード配線による論理回路により実行される。これは通常、機械サイクルの後半の間及び次の機械サイクルの第一の四分の一の間に行われる。ALUの出力は、第4図(k)の波形114の立ち上がり縁112に相当する次の機械サイクルの第二の四分の一の立ち上がり縁まで遅延される宛先のなかへロードされない。線100上のALU出力信号に対する宛先はレジスタファイル74のなかのレジスタ、命令カウンタ32又はア

キュムレータ140であってよい。アキュムレータは、シフト、乗算及び除算を実行する目的で設けられており、また、ALU演算の後でALU出力信号をレジスタファイルにロードする場合のように、選択された機械サイクルの間にCK2の立ち上がり縁でALUの出力をロードされる。

制御装置30はシステム受信器を更新し、また例外プログラムフロー（割込み、呼出し、飛越し）をチェックする。適当な制御信号がこのような場合に第3図中に示されている種々の機能エンティティに与えられる。

次に第5図を参照すると、第3図のレジスタファイル74が一層詳細に示されている。内部データバス102は内部データバス102上の信号と線100上のALU出力信号との間を選択する第一の3:1マルチプレクサ150及び第二の3:1マルチプレクサ152に導かれている。線100は、第5図では、ALU出力信号の最上位の半分を導く信号線100aとその最下位の半分を導く信号線100bとに分割されている。これらの

線100a及び100bの各々は第一及び第二のマルチプレクサ150、152の双方に導かれている。第3図の制御装置30は、どのマルチプレクサ150、152及びどの経路100、102がレジスタに信号を与えるかを制御する。

レジスタファイル74自体は20個の汎用レジスタの群を含んでいる。レジスタファイルは信号線100からのALUデータもしくは（データバス18から出発する）信号線102からのオペランドデータをロードされる。RISC命令はこれらのレジスタ上で演算を実行する。第5図には、二つの群に分割されたレジスタが示されており、第一の（“偶数”）群74a（R₀〜R₁₄ & A₀ & A₂）は第一のマルチプレクサ150からの線154上の出力信号に回答し、また10個の汎用レジスタを含んでいる。汎用レジスタ74bの第二の（“奇数”）群（R₁〜R₁₅ & A₁ & A₃）は第二のマルチプレクサ152からの線156上の第二の信号に回答する。いずれかの群74a、74bのなかのレジスタのいずれかがその内

容を信号線162、164を経て第三のマルチプレクサ158又は第四のマルチプレクサ160に供給し得る。第三のマルチプレクサ158は出力信号166を第3図の宛先レジスタ76に供給するための20:1マルチプレクサである。第四のマルチプレクサ160は線168上の出力信号を第3図の源レジスタ78に供給するための20:1マルチプレクサである。

第3図の実施例では、ALUは32ビットALUであるが、データ及びデータアドレスバス18、16も命令及び命令アドレスバス14、12も16ビットである。従って、第5図の構造に対しては、内部データバス102は16ビットであり、ALU信号出力線100aの最上位の半分の信号も線100b上の最下位の半分の信号も16ビットである。従って、マルチプレクサ出力線154、156も汎用レジスタ74a、74bの全ても16ビットである。20:1マルチプレクサ158、160の各々は32ビット語を構成し、最上位の半分は群74bのなかの10個のレジスタ

の一つから取られ、また最下位の半分は群74aもしくは74bのなかのレジスタのいずれか一つから取られる。こうして、宛先及び源レジスタ出力信号166、168は32ビット語である。

次に第6図を参照すると、第3図中に示されている源モジュール78のような源モジュールがブロック図で示されている。線168上の信号は32ビット信号であり、線176上の上位の16ビット及び線178上の下位の半分としてそれぞれ第一の3:1マルチプレクサ180及び第二の3:1マルチプレクサ182に与えるため点174で二つの16ビット語に分割されている。点184で線168上の32ビット語の最下位の5ビットは線186を経て、全てのビット演算に対する適当なビットを選択するビットデコーダ188に与えられる(ビットは変更される)。ビットデコーダ188からの32ビット語は、それぞれ第一及び第二のマルチプレクサ180、182に与えるため、線192上の最上位の半分と線194上の最下位の半分とに分割される。第一のマルチプ

レクサ180はレジスタファイルの最上位の半分の出力を選択し、又はビット命令上のビットフィールドを選択する。それはレジスタファイルのビット16出力を符号拡張する(全て1又は0に強制する)。第二のマルチプレクサ182はレジスタファイルの最下位の半分の出力を選択し、又はビット命令上のビットフィールドを選択する。それはバイトスワップ命令上で下位の8ビット及び上位の8ビットをスワップさせる。

線168a上の選択された最上位の半分及び線168b上の選択された最下位の半分は、第1図と結び付けて先に説明したようにALU演算の実行前に導き出された源データフィールドを保持するのに使用される源レジスタ78に与えられる。源はメモリへ又はそれからレジスタをロード又は記憶する以前にオペランドアドレスを与えるのにも使用される。これは第3図に示されており、出力線118がバッファ120に導かれており、バッファ120がオペランドアドレス信号を線16を経てオペランドメモリ空間に与える。前記のよ

うに、出力信号118は、もしオペランドアドレス指定機能を実行しないならば、第3図に一層良く示されているように、ALUに第一の入力信号128を与えるための命令線136とならんで、マルチプレクサ124に与えられる。

次に第7図を参照すると、第3図中の宛先モジュール76と類似の宛先モジュールがブロック図で示されている。レジスタファイル74のなかのマルチプレクサ158からの線166上の出力信号は、16及び32ビット語上の各クロック信号上の右シフト、左シフト又はノーシフトを許す32ビットのシフターマルチプレクサ200に与えられる。それは論理的、算術的及び循環的にシフトする。シフターマルチプレクサ200の出力は線202を経て宛先レジスタ204に与えられる。このレジスタはシフターマルチプレクサ200の導き出された宛先データフィールド出力を保持するのに使用される一時レジスタとして考えられ得る。宛先レジスタ204の出力は線136を経て、線130上の第二の入力信号を第3図のAL

U104に与えるためのマルチプレクサ126に与えられる。宛先モジュール76はメモリにレジスタを記憶する以前にオペランドデータを与えるためにも使用されている。宛先モジュールはクロックあたり1ビットだけ右又は左にシフトされ得るし、またシフト、乗算及び除算機能を実行するのに使用されている。

次に第8図を参照すると、第3図中のアキュムレータ140と類似のアキュムレータモジュール210がブロック図で示されている。線100上のALU出力信号は、線218上のアキュムレータレジスタ216の出力信号に回答する2:1マルチプレクサ212に与えられる。このマルチプレクサは32ビットの出力信号を線220を経て、乗算、除算及び64ビットシフトのために必要とされるような右シフト、左シフト又はノーシフトを許すシフターマルチプレクサ222に与える(それは宛先レジスタと連絡されている)。シフターマルチプレクサ222の出力は線224を経てアキュムレータレジスタ216に与えられる。

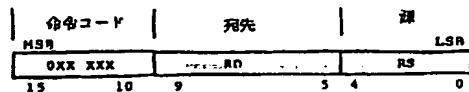
アキュムレータモジュール210は全体としてデータの一部記憶のためにも使用され得る。乗算及び除算の結果は宛先モジュール76及びアキュムレータモジュール140又は210のなかで定型化される。

第3図の信号プロセッサは16及び32ビット命令を支持する二つの基本命令フォーマットを有する。命令コード(opcode)は命令の6ビットの最上位ビットから成っている。第1表には32命令の命令コードマトリックスが示されている。命令コードの上側の2ビット及び次に下側の3ビットはそれぞれ、命令が置かれている行及び列を選択する。二つの命令フォーマットは(1)レジスタ・ツー・レジスタ(RR)及び(2)レジスタ・ツー・即値(RI)である。6ビット命令コードの最下位ビットは、それにより命令が解釈されるべきフォーマットを選択する。32命令の各々は第6ビットに関係して各フォーマットのなかで実行され得る。

第1表

	00	01	10	11
000	MOV	ADD	AND	SLL
001	LAR	ADDC	OR	SAR
010	STR	AB	XOR	SCR
011	CALL	ADDU	NOT	MULS
100	MOVC	SUB	RBR	MOVB
101	INR	SUBB	SBR	SWAB
110	OTR	SB	TBR	DIV
111	JCR	CHP	LRI	STRI

レジスタ・ツー・レジスタフォーマットは6ビットの命令コード及び二つの5ビットのレジスタフィールドから成る16ビットの命令である。レジスタフィールドは(1)20個の汎用レジスタ、(2)10個の汎用レジスタ対又は(3)1個のアキュムレータのいずれか一つを選択し得る。



レジスタ・ツー・レジスタ宛先(RD)及び源フィールド(RS)は第2表に従って選択され得る。

第2表

宛先及び源フィールド選択

ビット フィールド	RD が選択する 時	RS が選択する 時	ビット フィールド	RD が選択する 時	RS が選択する 時
00000	R0	R0	10000	XR0	XR0
00001	R1	R1	10001	XR2	XR2
00010	R2	R2	10010	XR4	XR4
00011	R3	R3	10011	XR6	XR6
00100	R4	R4	10100	XR8	XR8
00101	R5	R5	10101	XRA	XRA
00110	R6	R6	10110	XRC	XRC
00111	R7	R7	10111	XRE	XRE
01000	R8	R8	11000	A0	A0
01001	R9	R9	11001	A1	A1
01010	RA	RA	11010	A2	A2
01011	RB	RB	11011	A3	A3
01100	RC	RC	11100	XA0	XA0
01101	RD	RD	11101	XA2	XA2
01110	RE	RE	11110	—	*IMM
01111	RF	RF	11111	ACC	ACC

* * は源に対して16ビット拡張フィールドを使用する。

第3表には第5図のレジスタファイルの二つの半部に対して選択される組織が示されている。

第3表

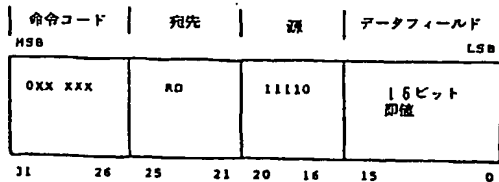
16ビット 16ビット レジスタ対

R0	R1	XR0
R2	R3	XR2
R4	R5	XR4
R6	R7	XR6
R8	R9	XR8
RA	RB	XRA
RC	RD	XRC
RE	RF	XRE
A0	A1	XA0
A2	A3	XA2
ACC		

表の左側の(16ビット)列は10個のレジスタ74aに相当し、表の右側の(16ビット)列は10個のレジスタ74bに相当する。前記のように、レジスタは32ビット語を形成するため対として作動するように選択され得る。レジスタ対に対する命名は第3表に示されており、またレジスタ・ツー・レジスタ命令のなかのレジスタフィールドによる選択のために第2表中に反映されて

いる。A 0 ~ A 3 レジスタは一般に中間結果を保持するために使用される。

レジスタ即値フォーマットは6ビットの命令コードと一つの5ビットのレジスタアドレスと即値命令を示す5ビットのコードと16ビットのデータフィールドとから成る32ビットの命令である。



レジスタ/アキュムレータ・ツリー即値フォーマットは命令コードマトリックスの命令を全て包含する。レジスタ/アキュムレータフィールドは第2表中に示されているRDフィールドとして選択され得る。

RISC命令集

アドレス モード	簡略記号	レジスタ 転送記号	影響される レジスタ
RR	MOV RD,RS	RD ← RS	***** RD
RR	LA RD,RS	IF (RS = SP) THEN RS ← RS + 1; AD ← @(RS); IF (RS != SP) THEN RS ← @(AD);	***** RD,SP
RR	STR RD,RS	IF (RS = SP) THEN RS ← RS - 1; RS ← @(RD); IF (RS != SP) THEN RS ← @(RD);	***** RS,SP
RR	CALL RD,RS	AD ← PC + 2 PC ← RS	***** PC, SP
RR	MOVC RD,RS	RD ← RS	OPIN RD, SW
RR	INR RD,RS	RD ← @(RS);	***** RD, SP
RR	OTR RD,RS	@(RS) ← RD;	***** SP

RR	JCR H,RS	IF (SH = H) THEN PC ← PC + RS; ELSE PC ← PC + 1;	***** PC
RR	ADD RD,RS	RD ← RD + RS;	CPINV RD, SW
RR	ADDC RD,RS	RD ← RD + RS + C;	CPINV RD, SW
RR	AB RD,RS	RD ₇₋₀ ← RD ₇₋₀ + RS ₇₋₀ ;	CPINV RD, SW
RR	ADDU RD,RS	RD ← RD + RS;	CPIN RD, SW
RR	SUB RD,RS	RD ← RD + RS + 1;	CPINV RD, SW
RR	SUBB RD,RS	RD ← RD + RS + C;	CPINV RD, SW
RR	SB RD,RS	RD ₇₋₀ ← RD ₇₋₀ + RS ₇₋₀ + 1;	CPINV RD, SW
RR	CMF RD,RS	RD : RS;	OPIN SW
RR	XOR RD,RS	RD ← RD ⊕ RS;	OPIN RD, SW
RR	NOT RD,RS	RD ← RS;	OPIN RD, SW
RR	ABR RD,RS	RD ← RD AND BIT(RS);	***** RD

RR	SBR RD,RS	RD ← RD AND BIT(RS);	***** RD
RR	TBR RD,RS	ALU ← RD AND BIT(RS);	OPIN SW
RR	LRI RD,RS	RD ← (RS);	***** RD
RR	SLR RD,RS	RD ← RD SHIFT (RS);	OPIN RD, SW
RR	SAR RD,RS	RD ← RD SHIFT (RS);	OPIN RD, SW
RR	SCR RD,RS	RD ← RD SHIFT (RS);	OPIN RD, SW
RR	MULS RD,RS	RD ← RD * RS;	***** RD, SW
RR	MOVB RD,RS	RD ₇₋₀ ← RS ₇₋₀ ;	***** RD
RR	SWAB RD,RS	RD ₁₅₋₈ ← RS ₇₋₀ ; RD ₇₋₀ ← RS ₁₅₋₈ ;	***** RD
RR	DIV RD,RS	RD ← RD / RS;	***** RD
RR	STRI RD,RS	@(RS) ← (RD);	***** none

命令集の詳細な説明

以下に説明されるRISC命令の各々はレジスタ・ツリーレジスタフォーマットに対して説明される。しかし、各々が命令コードの第6又は最下位ビットの状態により指定され得るレジスタ・ツリー即値フォーマットの等価物を有することは理解されよう。

移動命令(MOV)は源レジスタ(RS)の内容が宛先レジスタ(RD)のなかへ移動されることを許す。

ロードレジスタ命令(LR)はRSにより指示されたメモリ位置の内容をRDへ移動する。もしRSがスタックポインタ(SP)であれば、SPがロード以前にインCREMENTされる。

記憶レジスタ命令(STR)はRSにより指示されたメモリ位置のなかへRDの内容を記憶する。もしRSがスタックポインタ(SP)であれば、SPがデCREMENTされる。

呼出し命令(CALL)はRDにより指示されたレジスタのなかへプログラムカウンタの内容+2をロードする。RSの内容は次いでプログラムカウンタのなかへ移動される。

移動及び集合状態命令(MOVC)はRS及びRDの内容を移動する。

入力レジスタ命令(INR)はI/Oサイクルを規定する。RSにより指示されたI/O位置の内容をRDへ移動する。他方、もしレジスタ・ツ

ー・即値命令フォーマットがこの命令に対して規定されていれば、内部サイクルが指示され、それにより内部I/O位置がRDへ移動される。これらの内部位置の要約は下記の第4表に示されている。本明細書の現在の部分はRISCの説明に向けられているけれども、RISC、CISC又は本発明の第二の局面からRISC又は本発明の第一の局面を記述的に完全に分離することは往々困難であり、実施例は双方を組み合わせることは理解されよう。第4表中に開示される機能の多くはMIL-STD-1750により指定されており、従ってまた本発明の第一の局面に厳密に関係付けられない。しかし、それらは結合性のために本明細書にこの点で開示されている。しかし、それらはその目的で理解されべきである。

図4表

コマンドフィールド(RS)	記号	コマンド
(hex)	記号	コマンド
10	-	予約されている
11	EA0	有効アドレス 0
12	EA1	有効アドレス 1
13	EA2	有効アドレス 2
14	PI	保留部込み
15	MR	マスクレジスタ
16	FR	状態レジスタ
17	SW	状態レジスタ
18	IRS	命令レジスタ
19	IRD	命令レジスタ
1A	-	予約されている
1B	-	予約されている
1C	PC	プログラムカウンタ
1D	PIPE	パイプレジスタ
1E	PCREL	PC相対的
1F	-	予約されている

10. 予約されている。

11. 有効アドレス 0 (EA0) : RISCアーキテクチャにより実現されないMIL-STD-1750アドレス指定モード、すなわち直接及び直接指標付け(D、DX)アドレス指定モードを実現するために使用される。MIL-STD-1750命令レジスタ宛先フィールド(IR

D)により指示されるレジスタはパイプレジスタに加えられる。結果は選択されたアキュムレータ(A)のなかに記憶される。MIL-STD-1750パイプレジスタは次いでMIL-STD-1750プログラムカウンタにより指示されたメモリ位置の内容をロードされる。プログラムカウンタはポストインCREMENTされる。

12. 有効アドレス 1 (EA1) : RISCアーキテクチャにより実現されないMIL-STD-1750アドレス指定モード、すなわち基底相対(B)アドレス指定モードを実現するために使用される。MIL-STD-1750命令レジスタ基底フィールド(IRB)により指示されるレジスタは命令レジスタ(IR)の下位8ビットに加えられる。結果はAのなかに記憶される。

13. 有効アドレス 2 (EA2) : RISCアーキテクチャにより実現されないMIL-STD-1750アドレス指定モード、すなわち基底付き指標付き(BX)アドレス指定モードを実現するために使用される。MIL-STD-17

50 命令レジスタ基底フィールド (IRB) により指示されるレジスタはIRDフィールドにより指定されたレジスタに加えられる。結果はAのなかに記憶される。

14. 保留割込み (PI) を読む: MIL-STD-1750 により指令されるが、RISCもしくはCISCエミュレーションモードで使用される。保留割込みレジスタ (PI) の内容がAのなかに記憶される。

15. マスクレジスタ (MK) を読む: MIL-STD-1750 により指令されるが、RISCもしくはCISCエミュレーションモードで使用される。マスクレジスタ (MK) の内容がAのなかに記憶される。

16. 故障レジスタ (PT) を読む: MIL-STD-1750 により指令されるが、RISCもしくはCISCエミュレーションモードで使用される。故障レジスタ (PT) の内容がAのなかに記憶される。

17. 状態語 (SW) を読む: MIL-STD

-1750 モードに対して読み使用される。MIL-STD-1750 状態語レジスタ (SW) の内容がAのなかに記憶される。

18. 命令レジスタ源フィールド (IRS) を読む: MIL-STD-1750 プログラムレジスタ230の4ビットIRSフィールドがAのなかに記憶される。

19. IRDフィールド (IRD) を読む: MIL-STD-1750 プログラムレジスタ230のなかの4ビットIRDフィールドがAのなかに記憶される。

1A. 予約されている。

1B. 予約されている。

1C. プログラムカウンタ (PC) を読む: MIL-STD-1750 プログラムカウンタ234の内容がAのなかに記憶される。

1D. MIL-STD-1750 命令パイプレジスタ (PIPE) を読む: MIL-STD-1750 命令パイプレジスタ232内容がAのなかに記憶される。このパイプレジスタは次いでMIL

L-STD-1750 プログラムカウンタにより指示されたメモリ位置の内容をロードされる。プログラムカウンタはポストインクリメントされる。

1E. プログラムカウンタ相対的 (PCREL) を読む: MIL-STD-1750 命令レジスタ230のなかの8ビットフィールドを取り、それをMIL-STD-1750 プログラムカウンタ234に加える。結果はAのなかに記憶される。

1F. 予約されている。

出力レジスタ命令 (OTR) はRSにより指示された1/0位置のなかへRDの内容を記憶する。これは1/0サイクルに相当する。しかし、もしレジスタ・ツー・即値命令フォーマットがこの命令に対して指示されるならば、内部サイクルが指示され、それにより内部1/0位置がRDへ移動される。これらの内部1/0位置の要約は下記の第5表に示されている。

第5表

コマンドフィールド (RS) (hex)	符号記号	コマンド
10	CLPT	故障レジスタをクリア
11	PCL	プログラムカウンタロード
12	ENBL	読み込みをイネーブル
13	DSBL	読み込みをディスエーブル
14	SPI	保留割込みをセット
15	SHK	マスクレジスタをセット
16	SPT	故障レジスタをセット
17	ESW	状態語をセット
1A	IRS	命令レジスタ源フィールド
19	IRD	命令レジスタ宛先フィールド
1A	INCS	IRSフィールドをインクリメント
1B	INCD	IRDフィールドをインクリメント
1C	IRL	命令レジスタをロード
1D	IRLD	命令レジスタをロード
1E	CCOFF	条件コードをディスエーブル
1F	RPI	保留割込みをリセット

10. 故障レジスタをクリア (CLPT): RISCもしくはMIL-STD-1750 エミュレーションモードで使用し得る。16ビットの故障レジスタの内容が0にリセットされる。

11. プログラムカウンタをロード (PCL) : MIL-STD-1750 プログラムカウンタ 234 が A の内容をロードされる。

12. 割込みをイネーブル (ENBL) : MIL-STD-1750 により指令されるが、RISC もしくは CISC エミュレーションモードで使用される。このコマンドはマスクアウトされていない全ての割込みをイネーブルする。

13. 割込みをディスエーブル (DSBL) : MIL-STD-1750 により指令されるが、RISC もしくは CISC エミュレーションモードで使用される。このコマンドは DSBL 命令の実行の開始時に (ディスエーブルされ得ないように定義されているものを例外として) 全ての割込みをイネーブルする。一般に INT0 - 電源喪失、INT1 - 機械エラー、及び INT5 - 実行呼出しがディスエーブルされ得ない唯三つの割込みである。

14. 保留割込みレジスタをセット (SPI) : MIL-STD-1750 により指令されるが

SC もしくは CISC エミュレーションモードで使用される。故障レジスタ (FT) の内容は A の内容をロードされる。"1" のビット値は特定の故障をセットする。故障レジスタの必要条件に対するセクション 4 を参照。

17. 状態語をセット (SSW) : MIL-STD-1750 状態語 (SW) が A の内容をロードされる。

18. IRS (命令レジスタ宛先フィールド) フィールドをロード (IRS) : MIL-STD-1750 命令レジスタ 230 の 4 ビットの IRS フィールドが A のビット 5 ~ 8 をロードされる。

19. IRD (命令レジスタ宛先フィールド) フィールドをロード (IRD) : MIL-STD-1750 命令レジスタ 230 の 4 ビットの IRD フィールドが A のビット 1 ~ 4 (ビット 1 は LSB) をロードされる。

1A. IRS フィールドをインCREMENT (INCS) : MIL-STD-1750 の PR 230 のなかの 4 ビットの IRS フィールドがイン

CREMENT される。RISC もしくは CISC エミュレーションモードで使用される。このコマンドは A の 16 ビットの内容を保留割込みレジスタへ出力する。もし割込みマスクの対応するビット位置に "1" が存在し、(PI 及び MK の双方のなかの同一のビット集合)、且つ割込みがイネーブルされれば、割込みは次の命令の実行の後に生起する。

15. 割込みマスクレジスタをセット (SMK) : MIL-STD-1750 により指令されるが、RISC もしくは CISC エミュレーションモードで使用される。このコマンドは A の内容を割込みマスクレジスタへ転送する。対応するビット位置の "1" は割込みの生起を許し、また "0" は、マスクされ得ないものとして定義されているものを例外として、割込みの生起を阻止する。一般に INT0 - 電源喪失、INT1 - 機械エラー、及び INT5 - 実行呼出しがディスエーブルされ得ない唯三つの割込みである。

16. 故障レジスタをセット (SFT) : MIL-STD-1750 により指令されるが、R

LEMENT される。

1B. IRD フィールドをインCREMENT (INCD) : MIL-STD-1750 の PR 230 のなかの 4 ビットの IRD フィールドがインCREMENT される。

1C. 命令レジスタをロード (IRL) : MIL-STD-1750 のプログラムレジスタ 230 がパイプレジスタ 232 の内容をロードされる。RISC プログラムフローが (もし割込みが保留中であれば) 割込みベクトルに、もしくは (MIL-STD-1750 の PR 230 の上位 8 ビットにより規定される) マップベクトルに転送される。MIL-STD-1750 のパイプレジスタ 230 が次いで MIL-STD-1750 のプログラムカウンタ 234 により指示されたメモリ位置の内容をロードされる。プログラムカウンタが 234 ポスト・インCREMENT される。条件付き状態フィールドがイネーブルされる。

1D. 命令レジスタをロード (IRLD) : MIL-STD-1750 のプログラムレジスタ 2

30がパイプレジスタ232の内容をロードされる。RISCプログラムフローが(MIL-STD-1750のPR230の上位8ビットにより規定される)マップベクトルに転送される。MIL-STD-1750のパイプレジスタ230が次いでMIL-STD-1750のプログラムカウンタ234により指示されたメモリ位置の内容をロードされる。プログラムカウンタ234がポスト・インクレメントされる。条件付き状態フィールドがイネーブルされる。

1E. 条件コードをディスエーブル(CCOFF) : MIL-STD-1750の状態語(SW)の条件付き状態フィールドが、変更され得ないように、ディスエーブルされる。

1F. 保留割込みをリセット(RPI) : MIL-STD-1750により指令されるが、RISCもしくはCISCエミュレーションモードで使用される。Aのなかのセットされた各ビットに対して、その対応する割込みビットがリセットされる。

第6表

CCフィールド	飛越し条件	符号記号
NOP	...	
00001	0より小さい	JC LT,RS
00010	0に等しい	JC EQ,RS
00011	0より小さい/0に等しい	JC LE,RS
00100	0より大きい	JC GT,RS
00101	0に等しくない	JC NE,RS
00110	0より大きい/0に等しい	JC GE,RS
00111	無条件	...
10111	桁上げセット	JC CY,RS
01001	桁上げ又はLT	JC CLT,RS
01010	桁上げ又はEQ	JC CEQ,RS
01011	桁上げ又はLE	JC CLE,RS
01100	桁上げ又はGT	JC CGT,RS
01101	桁上げ又はNE	JC CHE,RS
01110	桁上げ又はGE	JC CGE,RS
01111	無条件	...
10000	あふれセット	JC V,RS
10001	あふれ又はLT0	JC VLT,RS
10010	あふれ又はEQ0	JC VE,RS
10011	あふれ又はLE0	JC VLE,RS
10100	あふれ又はGT0	JC VGT,RS
10101	あふれ又はNE0	JC VNE,RS
10110	あふれ又はGE0	JC VGE,RS
10111	無条件	...
11000	あふれ又は桁上げセット	JC VC,RS
11001	あふれ又は桁上げ又はLT0	JC VCLT,RS
11010	あふれ又は桁上げ又はEQ0	JC VCEQ,RS
11011	あふれ又は桁上げ又はLE0	JC VCLE,RS
11100	あふれ又は桁上げ又はGT0	JC VCGT,RS
11101	あふれ又は桁上げ又はNE0	JC VCHE,RS
11110	あふれ又は桁上げ又はGE0	JC VCGE,RS
11111	無条件	...

飛越し・条件付きレジスタ命令(JCR)は条件付き飛越し命令であり、もしCCフィールドに対応する論理"1"パターンがCCフィールドとCSフィールドとのビット・フェア・ビットのAND演算の結果であるならば、即値フィールドがプログラムカウンタに加算される。5ビットの条件状態フィールドは参照符号"VCPZN"を付されており、ここでVはあふれ、Cは桁上げ、Pは正、Zは0、Nは負を意味する。それらはALUによりセット又はリセットされる。CCフィールドはRISC命令レジスタ34のRSフィールドである。個々のCCコマンドが第6表に示されている。

加算レジスタ命令(ADD)はRSの内容をRDの内容に加える。結果はRDのなかに記憶される。あふれ条件は、もしオペランドが同一の符号を有し且つ結果が反対の符号を有するならば、生起する。

桁上げ付き加算レジスタ命令(ADDC)はRSの内容をRDの内容に加える。結果は、もし桁上げフラッグがセットされているならば、インクレメントされる。結果はRDのなかに記憶される。あふれ条件は、もしオペランドが同一の符号を有し且つ結果が反対の符号を有するならば、生起する。

加算バイト命令(AB)はRSの下位バイト内容をRDの下位バイト内容に加える。バイト結果はRDの下位バイトのなかに記憶される。あふれ条件は、もしバイトオペランドが同一の符号を有し且つバイト結果が反対の符号を有するならば、生起する。

加算レジスタ無符号命令(ADDU)はRSの内容をRDの内容に加える。結果はRDのなかに

記憶される。あふれ条件は影響されない。

減算レジスタ命令 (SUB) は RS の内容を RD の内容から差し引く。結果は RD のなかに記憶される。あふれ条件は、もしオペランドが反対の符号を有し且つ結果が RS と同一の符号を有するならば、生起する。

借り付き減算レジスタ命令 (SUBB) は RS の内容を RD の内容から差し引く。結果は、もし桁上げフラッグがクリアされているならば、デクレメントされる。結果は RD のなかに記憶される。あふれ条件は、もしオペランドが反対の符号を有し且つ結果が RS と同一の符号を有するならば、生起する。

減算バイト命令 (SB) は RS の下位バイト内容を RD の下位バイト内容から差し引く。バイト結果は RD の下位バイトのなかに記憶される。あふれ条件は、もしバイトオペランドが反対の符号を有し且つバイト結果が RS と同一の符号を有するならば、生起する。

比較レジスタ命令 (CMP) は RS の内容を R

D の内容と比較する。もし RD が RS よりも大きいならば、P 条件コードがセットされる。もし $RD - RS$ であれば、Z 条件コードがセットされる。もし RD が RS よりも小さいならば、P 条件コードがセットされる。

論理アンドレジスタ命令 (AND) は RS の内容と RD の内容とのアンド演算をする。結果は RD のなかに記憶される。

論理オアレジスタ命令 (OR) は RS の内容と RD の内容とのオア演算をする。結果は RD のなかに記憶される。

論理排他的オアレジスタ命令 (XOR) は RS の内容と RD の内容との排他的オア演算をする。結果は RD のなかに記憶される。

論理否定レジスタ命令 (NOT) は RS の内容の 1 の補数を RD のなかに記憶する。

レジスタ内ビットリセット命令 (RBR) は、RS のなかの値に従ってリセット (クリア) されるべき RD のなかのビットを選択する。下記の第 7 表には、RS のなかの値と RD のなかでクリア

されるビットとの関係が示されている。RS のなかの最下位 5 ビットが、クリアされるべきビットを決定するのに使用される。他のビット値は重要でない。

第 7 表

RS 内の値 MSB LSB	RS 内でクリアされるビット
0000 0000 0000 0000 0000 0000 0000 0001 0000 0000 0000 0010	31 MSB 30 29
0000 0000 0001 1101 0000 0000 0001 1110 0000 0000 0001 1111	2 1 0 LSB

レジスタ内ビットセット命令 (SBR) は、RS のなかの値に従ってセットされるべき RD のなかのビットを選択する。下記の第 8 表には、RS のなかの値と RD のなかでセットされるビットとの関係が示されている。RS のなかの最下位 5 ビットが、セットされるべきビットを決定するのに使用される。他のビット値は重要でない。

第 8 表

RS 内の値 MSB LSB	RD 内でセットされるビット
0000 0000 0000 0000 0000 0000 0000 0001 0000 0000 0000 0010	31 MSB 30 29
0000 0000 0001 1101 0000 0000 0001 1110 0000 0000 0001 1111	2 1 0 LSB

レジスタ内ビットテスト命令 (TBR) は、RS のなかの値に従ってテストされるべき RD のなかのビットを選択する。下記の第 9 表には、RS のなかの値と RD のなかでテストされるビットとの関係が示されている。RS のなかの最下位 5 ビットが、テストされるべきビットを決定するのに使用される。他のビット値は重要でない。もしテストされるべきビットが 0 であれば、状態語のなかの 2 ビットがセットされる。

第9表

RS内の値		RD内でテストされるビット	
MSB	LSB		
0000 0000 0000 0000		31	MSB
0000 0000 0000 0001		30	
0000 0000 0000 0010		29	
0000 0000 0001 1101		2	
0000 0000 0001 1110		1	
0000 0000 0001 1111		0	LSB

ロード命令レジスタ命令 (LRI) はRSにより指示された命令メモリ位置の内容をRDのなかに記憶する。

シフト論理レジスタ命令 (SLR) はRSの内容により選択されたビットの数だけRDの内容を論理的にシフトする。RSの下位5ビットがシフトカウント及び方向を選択する。従って、シフト操作のために、可能なシフトの最大数は32である。RSの6ビットは通常又は拡張シフトを選択する。それがRSのビット16と反対の符号である時、拡張シフトモードが選択される。拡張シフ

シフト算術レジスタ命令 (SAR) はRSの内容により選択されたビットの数だけRDの内容を算術的にシフトする。もしRDの符号が左シフトの間に変化すれば、あふれが生起する。RSの下位5ビットはシフトカウント及び方向を選択する。従って、シフト操作のために、可能なシフトの最大数は32である。RSの6ビットは通常又は拡張シフトを選択する。それがRSのビット16と反対の符号である時、拡張シフトモードが選択される。拡張シフトモードでは、一時レジスタ (TA) はシフトされるべき値の上位32ビットを含み、またレジスタ対RDは下位32ビットを含む。拡張シフトを使用して、64ビットがシフトされ得るが、命令あたりは32回のみである。シフト論理レジスタ命令 (SLR) と結び付けて示された第10表は、SAR命令に対して、実行すべきシフトの方向及び数を決定する時にRSの値をどのように選択するかを示すのにも使用され得る。

シフトサイクルレジスタ命令 (SCR) はRS

トモードでは、一時レジスタ (TA) はシフトされるべき値の上位32ビットを含み、またレジスタ対RDは下位32ビットを含む。拡張シフトを使用して、64ビットがシフトされ得るが、命令あたりは32回のみである。下記の第10表には、実行すべきシフトの方向及び数を決定する時にRSの値をどのように選択するかを示されている。

第10表

方向	通常 (RS) hex	シフトの数	拡張 (RS) hex	シフトの数
左	0000	1	0020	1
	0001	2	0021	2
	0016	31	003E	31
右	0017	32	003F	32
	FF00	32	FFC0	32
	FF01	31	FFC1	31
	FFFE	2	FFDE	2
	FFFF	1	FFDF	1

の内容により選択されたビットの数だけRDの内容を回転する。RSの下位5ビットはシフトカウント及び方向を選択する。従って、シフト操作のために、可能なシフトの最大数は32である。RSの6ビットは通常又は拡張シフトを選択する。それがRSのビット16と反対の符号である時、拡張シフトモードが選択される。拡張シフトモードでは、一時レジスタ (TA) はシフトされるべき値の上位32ビットを含み、またレジスタ対RDは下位32ビットを含む。拡張シフトを使用して、64ビットがシフトされ得るが、命令あたりは32回のみである。SLRと結び付けて示された第10表は、SCR命令に対して、実行すべきシフトの方向及び数を決定する時にRSの値をどのように選択するかを示すのにも使用され得る。

乗算レジスタ命令 (MULS) はRSの内容によりRDの内容を乗算する。

移動レジスタ内バイト命令 (MOVB) はRSの下位バイト内容をRDの下位バイト内容のなかへロードする。

スワップレジスタ内バイト命令 (SWAB) は RS の下位及び上位バイト内容をそれぞれ RD の上位及び下位バイト内容のなかへロードする。

除算レジスタ命令 (DIV) は RS の内容により RD の内容を除算する。比は RD のなかに記憶される。残余は TA のなかに記憶される。

記憶レジスタ命令コード命令 (STRI) は RS により指示された命令メモリ位置のなかへ RD の内容をロードする。これは命令メモリサイクルである。

RISC CISC

再び第3図を参照すると、状態括 (SW) レジスタ、保留割込み (PI) レジスタ、割込みマスク (MK) 及び故障レジスタ (PT) の形態でシステム状態を含んでいる CISC に対する追加的な特徴が I/O 110 により得られることが思い出されよう。これらは全て MIL 排他的オア回路 STD 排他的オア回路 1750 により定められている。RISC と結び付けて先に説明したように、それはオペランドデータをレジスタファイルに

選択するのに使用されたことは思い出されよう。それは、CISC エミュレーションモードで、信号線 238 を経てレジスタファイルに CISC プログラム命令を連係する追加的な特徴を提供する。同様に、マルチプレクサ 126 は信号線 240 を経てレジスタファイルに CISC プログラムアドレスを連係するのに使用される。

任意の所与の CISC 応用プログラムに対して、プログラムカウンタ 234 は、類似の仕方での実行のためにパイプ 232 のなかに受信される各順次 CISC 命令を通じてインCREMENTする。各 CISC 命令が受信されるにつれて、それは線 242 を経て制御装置 30a に与えられ、それにより復号される。制御装置は、RISC 命令の群のなか第一の RISC 命令が線 242 を経て受信された特定の CISC 命令をエミュレートするため第2図の PROM 20 のなかに記憶されている対応する RISC アドレスを "ルックアップ" する。この RISC の第一のアドレスが信号線 244 上に置かれ、また RISC 命令カウンタ 32 に

連係するのに使用される。エミュレーションのために使用される RISC に対して、CISC プログラムレジスタ 230 が、現在実行されている CISC プログラム命令を保持するのに使用される一時レジスタとして設けられている。これらの命令は CISC 命令 "パイプ" 232 から受信される。このパイプ 232 は、CISC プログラムカウンタによりアドレス指定されてバッファ 106、信号線 108 及び I/O 110 を経て与えられる CISC 命令をデータバス 18 を経て受信する。CISC プログラム命令アドレスは、対応する CISC プログラム命令をパイプ 232 のなかへロードする以前に CISC プログラムレジスタ 234 のなかに一時記憶される。プログラムカウンタはその CISC 命令アドレスを、制御装置 30a により制御されるバッファ 236 を経てデータバス 16 上に与える。

基本 RISC のマルチプレクサ 124 が、算術及び論理演算のため及びレジスタファイルへの命令データの連係のために ALU への第一の入力を

与えられ、そこから次いで RISC 命令アドレスバス 12 上に置かれる。所望の RISC 命令が次いで命令バス 14 上に現れ、また前記のように復号且つ実行される。命令カウンタは次いでこの機械サイクルの中央で、すなわち制御装置 30a により制御されてエミュレーション群の開始時に RISC 命令をアドレス指定するため CK3 の立ち上がり縁で INCREMENT される。クロック線 246 は、CK3 の立ち上がり縁で高レベルになる信号線を示す。命令カウンタはエミュレーション群の第一の命令以外の全ての命令に対して CK1 で INCREMENT される。

群のなかの第一の RISC エミュレーション命令で命令カウンタをロードする以前の機械サイクルの間、パイプ 232 及びプログラムレジスタ 230 は、それらのそれぞれの内容を適当な宛先に、すなわちプログラムレジスタ 230 の内容を制御装置 30a に、またパイプ 232 の内容を (もし適当であれば) マルチプレクサ 124 に与える目的で CK1 の立ち上がり縁でクロックされてい

る。すなわちこのCKIの立ち上がり縁でプログラムカウンタ234がクロックされる。こうして、特定の機械サイクルの間にCISC22のなかへクロックされる新しいCISC命令に対して、対応する最初のRISC命令アドレスがRISC命令カウンタ32のなかへロードされる以前に半サイクルの遅延が存在する。

基本RISCと結び付けて以上に説明された命令集合を使用してこの仕方で行われるCISCの例は後で示される。追加的な例は不必要である。なぜならば、熟練したプログラマーは、CISC命令の集合を与えられて、RISCエミュレーション群の対応する集合を容易に構成し得るからである。

例として選択されるCISC命令集合は、命令コードの一つが直接アドレス指定モードのなかの単精度整数加算であるMIL-STD-1750 A命令集合である(1980年7月2日付けのMIL-STD-1750Aの第89頁の5.55節を参照)。そのモードのなかで、加算のためには

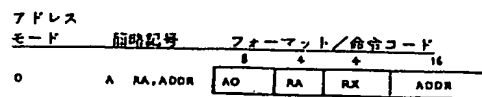
位16ビット(ADDR)はバイブ232に与えられる。もちろん、示されている実施例では、これらの命令が受信されるオペラントデータバスは単に16ビットのバスであり、上半及び下半が別々に送られなければならないことは理解されよう。いずれの場合にも、プログラムレジスタ230のなかの上半は線242を経て制御装置30に与えられ、そこで復号され、また特定のCISC命令をエミュレートするためRISCエミュレーション群の開始アドレスを示す適当なRISCアドレスが線244を経てRISC命令カウンタ32に与えられる。

MIL-STD-1750Aの単精度整数加算をエミュレートするための第一のこのようなRISC命令は下記の通りである:

INR A0, EAI

上記の命令に従って、バイブ232のなかに残っている下位16ビットのアドレスが次いでマルチプレクサ124及びALUI04を経てレジスタファイル74のレジスタA0のなかへ送られ

、導き出されたオペラントがRAレジスタの内容に加算される。結果(2の補数)はレジスタAのなかへ記憶される。フォーマットは下に示されている。



8ビットの命令コードは“加算直接”演算コードである。4ビットのRAフィールドは、導き出されたオペラントが加算され且つ結果が記憶されるレジスタを指定する。RAフィールド(4ビット)はインデックスレジスタである。16ビットの加算器フィールドは導き出されたオペラントを得るために設けられている。こうして、示されている命令は、第2図の命令バス24がアクセスしたMIL-STD-1750によるメモリ空間のなかへ記憶され得る。それはデータバス18を経て与えられ、また上位16ビットは復号のために線242を経て制御装置30に与えるためプログラムレジスタ230のなかへ与えられ、また下

る。

命令カウンタ32が次いで下記のような次のエミュレーション命令をアドレス指定するためインクレメントされる:

LR A1, A0

上記のロードレジスタ命令は、A0のなかに残っているアドレスにより指示されるメモリ位置の内容がA1へ移動されるメモリサイクルである。この群のなかで実行されるべき次のRISCエミュレーション命令は下記の通りである:

ADD IRS, A1

上記の加算レジスタ命令はPR*フィールド(4ビット)により指示されたレジスタの内容を取り、またそれをレジスタA1の内容に加える。加算の結果はPR*フィールド(16のR*レジスタの一つ)により指示されたレジスタのなかへ記憶される。この群に対する次の最後のRISCエミュレーション命令は、基本的に言って、バイブ232及びレジスタ230からの次のCISC命令を取る。それは下記の通りである:

O T R A 0 . I R L

こうして、M I L - S T D - 1 7 5 0 A の命令の各々が第 2 図の P R O M 2 0 のなかに記憶されている R I S C 命令の群としてエミュレートされ得る。プログラミングの分野の当業者にとって、それぞれ複合命令集合の命令の一つをエミュレートするための R I S C 命令の種々の群を構成することは簡単である。従って、R I S C コードでエミュレートされる全ての M I L - S T D - 1 7 5 0 A の命令の例はここには示されない。示された例で十分である。

M I L - S T D - 1 7 5 0 C I S C をエミュレートするための R I S C エミュレータを設計するために本アプローチを使用することにより特別な追加的な利点が得られる。第 2 図に示されている 2 チップ解決では、P R O M 2 0 がユーザー固有の特別なユーザーにより定義された機能を定義するのにユーザーにより使用され得る。1986 年 9 月 29 日付けの M I L - S T D - 1 7 5 0 B のドラフトバージョンの第 147 頁には、「特殊」

してユーザー固有のビルトイン機能を非常に安価に製造し得る。これは従来のアプローチを使用しては可能でなかった非常に望ましい特徴である。

設計方法

本発明の第三の局面によれば、複合命令集合計算機 (C I S C) アーキテクチャを実現するための設計方法が提供される。C I S C を設計する現在の方法はいくつかの異なる形態をとる。第一のアプローチでは、単一レベル制御を使用して全ての命令が実現される。換言すれば、実行は例えば 28000 に使用されているようなハードウェアにより制御される。第 9 図には、C I S C 命令集合アーキテクチャ 300 が「ハード配線による」制御装置 302 を介して実現されるこのようなアプローチが示されている。

他のアプローチでは、2 レベル制御を使用して全ての命令が実現される。換言すれば、実行はハードウェアのマикроコード (ファームウェア) 制御により制御される。第 10 図には、信号線 306 上の命令を介してファームウェア 308 へ供

アドレス指定モードに対するビルトイン機能命令が記載されている。この命令はユーザーにより定義された特殊な演算を呼び出す。命令はそれにすぐ続く一つ又はそれ以上の追加語を使用し得る。その数及び解釈は 8 ビットの命令コード拡張により決定される。市販品の M I L - S T D - 1 7 5 0 チップに対してこのようなビルトイン機能を与えるための現在のアプローチは、特別に要求されるビルトイン機能を組み入れるため顧客ノイドのアプローチに従ってシリコン上に縮小されなければならない製造者からの特製チップを顧客が特別に注文することである。これは非常に費用がかかるアプローチであり、費用が制約されている多くの応用にはオプションであり得ない。しかし、本発明のアプローチを使用すれば、ユーザーは R I S C 命令集合の可能性を知らされ、またユーザー固有のビルトイン機能を P R O M 2 0 に格納するためのソフトウェアのなかにプログラムし得る。こうしてユーザーは、R I S C ハードウェア 10 のなかで実行される R I S C ソフトウェアを使用

給される C I S C 命令集合アーキテクチャ 304 が示されており、ファームウェア 308 が残余の実行機能を信号線 312 を経てハードウェア制御装置 310 へ通す。このようなアプローチの例はモトローラ 68000 である。

別のアプローチでは、たいていの命令は 2 レベル制御を使用して実行され、残余の命令はソフトウェアを使用して実行される。このアプローチは第 11 図に示されており、C I S C 命令集合アーキテクチャ 314 に属する複合命令はファームウェア制御 316 とソフトウェアエミュレーション 318 を供給するための手段との双方に供給される。ファームウェアは信号線 320 上の複合命令とエミュレーションユニット 318 で行われたソフトウェアエミュレーションの結果を示す線 322 上の信号とに応答する。残余の制御機能はハード配線による制御装置 324 のなかで実行される。このような実現の例はマイクログラフィックスである。

第 12 図には、複合命令集合計算機アーキテク

チュアを実現するための本発明による方法が示されている。このアプローチでは全ての命令が2レベル制御を使用して実行され、実行はハードウェアのソフトウェア制御により制御される。例はRISCによりエミュレートされるMIL-STD-1750AのCISCに対して先に示されている。ソフトウェアエミュレーションは、従来の場合のように、ソフトウェアインタプリタを必要とする。ソフトウェア解釈は遅い。さらに、全ての命令は実行され得ない。ここに開示される方法はMIL-STD-1750、VAX、NEBULAなどを含む多数のCISC命令集合に適用される。このアプローチでは、まずRISC設計フィロソフィを使用して（ハード配線による）単レベル制御が構成される。そうする間に、設計者はRISC（ハード配線）命令集合の実行を最大化するべく試みる。いったんRISCがハードウェア設計されると、それはシリコンへの縮小のために工場に送られ得る。設計者は次いで、上記の例で説明したように、RISC命令を使用してCISC

エミュレータを書く。このアプローチの合理性はRISC設計時間がCISC設計時間よりもはるかに少ないことである。例えば、フェアチャイルドF9450、MD281は開発に3年以上も要したことが知られている。本アプローチを使用すれば、MIL-STD-1750のRISCエミュレータは1年以内で開発され、また認可を得るのにシリコン工場への送付を1回しか必要としなかった。

第12図には本発明のアプローチを使用する設計方法が示されている。CISC命令集合アーキテクチャ326は信号線328を経て、ハード配線による装置334のなかでのハードウェア実行のために線323上に信号を供給するソフトウェアエミュレーションユニットへ複合命令を供給する。このアプローチの主要な利点は、(1)ソフトウェアの速度に対するハードウェアの複雑さのかわあい、及び(2)ハードウェアの設計時間（通常はソフトウェアの設計時間よりもはるかに長い時間を要する）の短縮である。こうして、最

初の設計からワーキングコントロールまでの時間ははるかに少なくすむ。

第13図は本発明の第三の局面による設計ステップを示すブロック図である。RISCは先ずステップ340で設計される。いったんハードウェア設計及びRISC命令集合が決められると、次のステップ342は製造のための設計に送られる。同時に、第2図のPROM20上で一緒に群にされているRISC命令で選択されたCISC命令集合をエミュレートするためエミュレーションコードが書かれるステップ344が同時に実行される。

以上に於ては本発明を特定の好ましい実施例について説明してきたが、本発明はこれらの実施例に限定されるものではなく、本発明の範囲内にて種々の実施例が可能であることは当業者にとって明らかであろう。

4. 図面の簡単な説明

第1図は本発明の第一の局面によるRISC10のブロック図である。

第2図は本発明の第二の局面によりデータバス18を経て複合命令の源から受信された複合命令をエミュレートするためのRISC命令の群を含んでいる別個のメモリストア20とならんで使用される本発明の第一の局面によるRISC10のブロック図である。

第3図は本発明の第一の局面によるRISC10を一層詳細に細線で示し、また本発明の第二の局面によりCISCをエミュレートし得るRISCを実行するために必要な追加的ハードウェアを太線で示すブロック図である。

第4図は第3図中に示されている信号線上の種々の信号の電圧レベルにを示す種々の波形図である。

第5図は第3図のレジスタファイル74を一層詳細に示すブロック図である。

第6図は第3図の源モジュール78を一層詳細に示すブロック図である。

第7図は第3図の宛先モジュール76を一層詳細に示すブロック図である。

第8図は第3図のアクムレータモジュール140を一層詳細に示すブロック図である。

第9図は全ての命令が単一レベル制御を使用して実行されるCISCの従来の設計実行を示すブロック図である。

第10図は全ての命令が二レベル制御を使用して実行されるCISCの設計への従来のアプローチを示すブロック図である。

第11図は大部分の命令が二レベル制御を使用して実行され、また残りの命令がソフトウェアを使用して実行されるCISCの設計への従来のアプローチを示すブロック図である。

第12図は本発明の第三の局面により全ての命令が二レベル制御を使用する設計アプローチを使用するCISC実行を示すブロック図である。

第13図は本発明の第三の局面によりCISCをエミュレートするためのRISCを設計・製造する際に連行され得るステップを示すブロック図である。

10…信号プロセッサ(RISC)、12…R

ソフトウェアエミュレーションユニット、334…ハード配線による制御装置

特許出願人 ユナイテッド・テクノロジーズ・コーポレーション

代理人 弁理士 明 石 昌 毅

S C命令アドレスバス、14…縮小命令バス、16…オペランドアドレスバス、18…データバス、20…メモリ(PROM)、30…制御装置、32…命令カウンタ、34…命令レジスタ、74…レジスタファイル、76…宛先モジュール、78…源モジュール、104…ALU、124、126…マルチプレクサ、140…アクムレータモジュール、150、152、158、160…マルチプレクサ、200…シフトマルチプレクサ、204…宛先レジスタ、210…アクムレータモジュール、212…マルチプレクサ、222…シフトマルチプレクサ、300…CISC命令集合アーキテクチャ、302…ハード配線による制御装置、304…CISC命令集合アーキテクチャ、308…ファームウェア制御ユニット、310…ハードウェア制御装置、314…CISC命令集合アーキテクチャ、316…ファームウェア制御ユニット、318…エミュレーション装置、324…ハードウェア制御装置、326…CISC命令集合アーキテクチャ、330…

図面の添付。

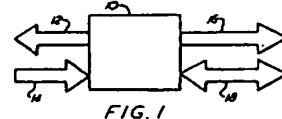


FIG. 1

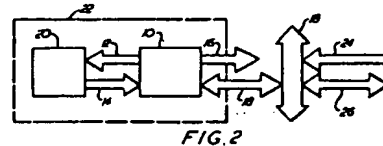


FIG. 2

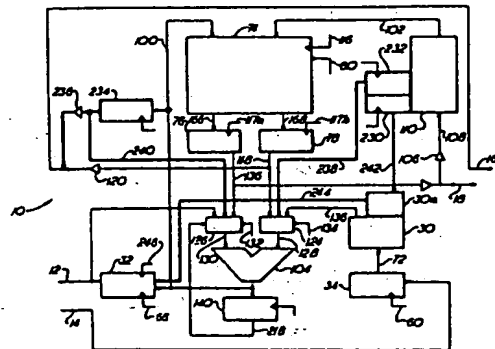
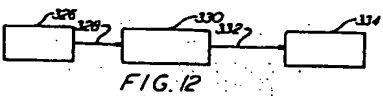
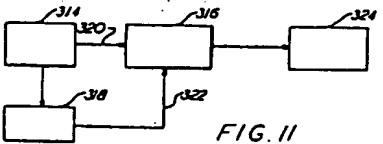
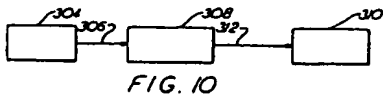
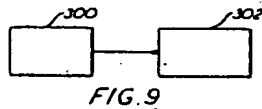
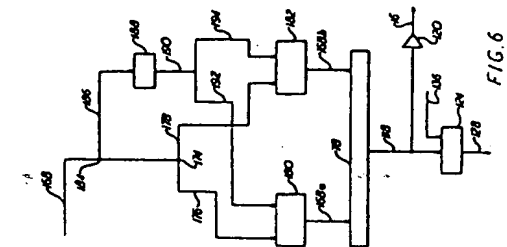
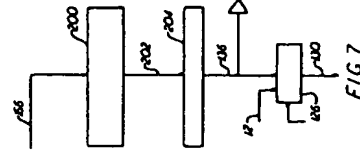
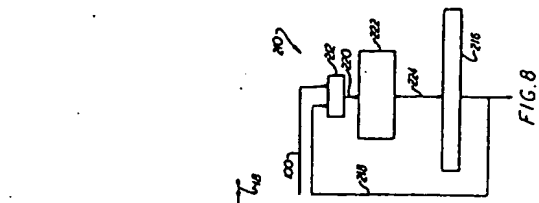
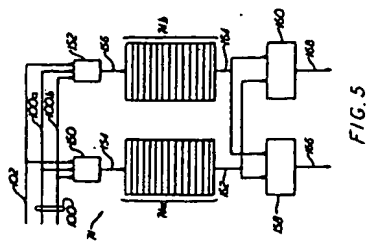
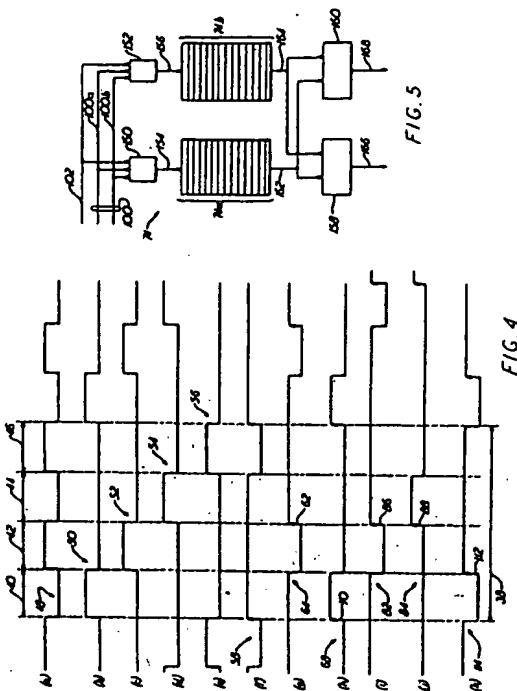


FIG. 3



**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.